International Handbooks on Information Systems

Series Editors
Peter Bernus · Jacek Błażewicz · Günter Schmidt · Michael Shaw

Springer-Verlag Berlin Heidelberg GmbH

Titles in the Series

P. Bernus, K. Mertins and G. Schmidt (Eds.)
Handbook on Architectures of Information Systems
ISBN 3-540-64453-9

M. Shaw, R. Blanning, T. Strader and A. Whinston (Eds.)
Handbook on Electronic Commerce
ISBN 3-540-65822-X

J. Błażewicz, K. Ecker, B. Plateau and D. Trystram (Eds.)
Handbook on Parallel and Distributed Processing
ISBN 3-540-66441-6

H. H. Adelsberger, B. Collis and J. M. Pawlowski (Eds.)
Handbook on Information Technologies for Education and Training
ISBN 3-540-67803-4

C. W. Holsapple (Ed.)
Handbook on Knowledge Management 1
Knowledge Matters
ISBN 3-540-43527-1
Handbook on Knowledge Management 2
Knowledge Directions
ISBN 3-540-43527-1

P. Bernus, L. Nemes and G. Schmidt (Eds.)
Handbook on Enterprise Architecture
ISBN 3-540-00343-6

Peter Bernus · Laszlo Nemes
Günter Schmidt
Editors

Handbook on Enterprise Architecture

With 237 Figures
and 23 Tables

Springer

Associate Professor Dr. Peter Bernus

Enterprise Integration Group
School of Computing and Information Technology
Faculty of Information and Communication Technology
Griffith University
Nathan (Brisbane) Queensland 4111, Australia
e-mail: bernus@cit.gu.edu.au

Dr. Laszlo Nemes

CSIRO Manufacturing & Infrastructure Division
Cnr Raglan and Albert Streets
Preston Vic, Locked Bag No. 9
Preston Vic 3072, Australia
e-mail: laszlo.nemes@csiro.au

Professor Dr. Günter Schmidt

University of Saarland
Information and Technology Management
Postfach 151150
66041 Saarbruecken, Germany
e-mail: gs@citm.uni-sb.de

ISBN 978-3-642-05566-9 ISBN 978-3-540-24744-9 (eBook)

DOI 10.1007/978-3-540-24744-9

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie;
detailed bibliographic data available in the internet at *http.//dnb.ddb.de*

http://www.springer.de

Originally published by Springer-Verlag Berlin Heidelberg New York in 2003
Softcover reprint of the hardcover 1st edition 2003

Cover design: Erich Kirchner, Heidelberg

SPIN 10908105 42/3130 - 5 4 3 2 1 0 - Printed on acid-free paper

Preface

We recommend this book as a practical guide and as a comprehensive volume for reference for *enterprise engineers.* We hope that business managers, ICT professionals, researchers, developers, consultants, government experts, and university students will keep it on a special place on their bookshelves as it fundamentally differs from many other works on this subject. Every part of it has proved to be useful in our practice in numerous projects. We have made parts of the manuscript available to universities as teaching material and to industrial partners for designing and managing their projects. Feedback from these sources has been great and comments and contributions have improved the content and readability of the material tremendously.

IT vendors, system builders, e-commerce companies, web-based companies, aerospace vendors, CASE tool and Enterprise Engineering Tool vendors will find this volume useful as a source of ideas and methods that they can apply both in the development of their own organisation and in the development of their products. It is highly recommended as a textbook for teaching enterprise integration at the Masters level at universities and for professional development in many organisations.

The intention has been to address professionals with wide variety of interest and with different level of knowledge or theoretical background. As a result, the person who reads its pages may not necessarily do so from cover to cover. Chapters will carry different significance to different individuals; therefore, they may well decide to study the sections in their preferred orders. We have tried hard to make each major section as much self-contained as possible without too much repetition. Reappearance of some material proved to be essential in order to avoid annoying cross-references. The readers of the manuscript reported this practice as being very useful and they underlined this with another reason: complex, sometimes abstract concepts can be more easily understood when they are explained in different contexts.

Let us give some ideas how this book can be read in many ways.

Readers	Chapters
This book offers a comprehensive guide for *Corporations, defence and government departments* to introduce enterprise integration capability and implementing a wide selection of change processes. It can be used to develop necessary skills and assemble tool sets for large organisations that wish to build a continuous change capability.	**4-10, 19-22, 2**
Consultants and engineers may turn to the sections explaining synthesis methods, and software tools. The presented methods and techniques and examples have a wide range of applicability – going beyond the techniques of any one discipline alone.	**4, 6-14, 15, 17-18, 2, 3**
Managers and project leaders need examples on how to keep the complexity of enterprise integration programmes under control and what the capabilities of the plethora of tools, building blocks and languages are.	**4-5, 8-18, 2**
Researchers and graduates will benefit from the comprehensive framework in which research problems and programmes can be characterised.	**2-4, 7, 10-15**
Business managers will use the methods to design innovative new business systems that are needed to take advantage of recently matured information and communication technologies.	**4-5, 7, 10, 12, 15-18, 19-22, 2**
IT vendors, system builders, e-commerce companies, aerospace vendors, CASE tool and enterprise engineering tool vendors will find the reference parts particularly useful.	**10-14, 17-18, 2, 3**
The book may be of special interest for *defence personnel* in capability development, defence acquisition and logistics	**4, 6-7, 10, 12, 18, 2, 3**

Editing this volume as a Handbook, we finally agreed to put the Chapters in the following order:

Chapter 2: GERAM (IFIP / IFAC Task Force)
Chapter 2 is the official document for the Generalised Enterprise Reference Architecture and Methodology (GERAM, Annex A to ISO/IS 15704). It is about those methods, models and tools, which are needed to build and maintain the integrated enterprise, be it a part of an enterprise, a single enterprise or a network of enterprises (virtual enterprise or extended enterprise).
GERAM defines a tool-kit of concepts for designing and maintaining enterprises for their entire life. It is not yet-another-proposal for an enterprise reference architecture, but is meant to organise existing enterprise integration knowledge. The framework has the potential for application to all types of

enterprise. Previously published reference architectures can keep their own identity, while identifying through GERAM their overlaps and complementing benefits compared to others.

Chapter 3: Reference Architecture Mapping (Noran)
Chapter 3 presents the mapping of selected life cycle enterprise architectures and architecture frameworks onto the GERAM framework. It covers aspects regarding the reference architectures (including life cycle / life history concepts and modelling frameworks) and associated modelling methodologies, languages and reference models where applicable. The Chapter builds on previous research efforts of mapping established frameworks, giving a refined interpretation of those results. Several emerging frameworks are also reviewed and mapped against the GERAM.
This Chapter is organised in accordance with the GERAM structure. As such, it can be an essential companion in choosing the necessary elements (modelling framework, methodology, tools etc) for the creation of a *complete*[1] enterprise model.

Chapter 4: Strategy as a Creation of Corporate Future (Kalpic et al)
Chapter 4 presents an introvert perspective of strategy formation, through the presentation of the resource-based view (RBV) concept. It reviews the relevant fundamentals of strategic management (definition of strategy, resources and capabilities), and strategy framework. The latter incorporates the conceptual knowledge of the RBV intertwined with some well-known tools in the domain of application.
The presented strategy framework is not intended to be a prescriptive step-by-step methodology; instead it offers guidance and reminds the user of issues to be considered in the process of the strategy formation.

Chapter 5: Leadership – Better Relationships through Better Communication (Mackay)
The significance of communication is highlighted in Chapter 5. There is a natural tendency to defend the *status quo* – if individuals in the organisation do not see that the arguments for change stand to reason. They naturally defend the present arrangements and keep doing things the same way they have been done in the past either through voicing opposition or by actions. Change implies abandoning routines, learning, negotiating, establishing new authorities and responsibilities and losing some old ones. All of these take people out of the comfort zone. As opposed to this no-change takes the least effort, and minimises short-term risk – so, even if things are not working perfectly people know how to deal with problems on the day-to-day level. Perhaps people can be convinced by pure logical arguments that change is necessary but the

[1] for the envisaged purpose and audience

tendency to avoid it remains very strong. *How can top management or any leader of a change effort negotiate such an obstacle?*

Chapter 6: Capability Improvement (Goranson)

Chapter 6 addresses two dimensions together, its focus being on the business mechanics and modelling challenges involved in evolving to this next generation of frameworks.

Enterprise modelling originated in the operational side of the enterprise and traditionally it was focused on that part of the enterprise with which operational managers concern themselves. This meant that the strategic planning of the enterprise, the marketing strategy, the design of a financial strategy to support the business model (indeed refinement of the business model itself), and major parts of the product design (if applicable) were presumed to have taken place before the enterprise modelling process began.

Enterprise models were built with the intent of 'engineering' the enterprise to optimally support these presumptions. It would then operate in a fairly stable mode for a long time. After some time, prompted by external changes and/or new insights or improvements in 'internal' processes, the enterprise would be 're-engineered' in order to operate in the new configuration for another long, stable period. The major problems with this approach are that high-risk processes are excluded, and dynamism within the enterprise is not well supported in response to new conditions or insights.

But we are now well into the evolution of enterprise integration via enterprise modelling into a new, 'second generation'. With this new generation of frameworks, strategic and key business infrastructure *is* incorporated into the enterprise-modelling framework. This will allow the 'balance' of operational decisions to be closely tied to strategic business goals. Evolution in another dimension will allow continuous engineered adjustment to the enterprise in response to dynamic conditions, rather than waiting for big 'batch' changes.

Chapter 7: Developing the Business Model (Tolle et al)

Chapter 7 presents a methodology to develop a *Business Model* of Virtual Enterprises. Enterprises co-operate with other enterprises in all phases of product life cycle to achieve cost reduction, increase their operational flexibility, and allow them to focus on core competencies. The preferred method for co-operation can be anything from long term alliances between partners in fixed supply chains – to a goal oriented, project focused co-operation as it is usually done in virtual enterprises (VEs).

In the initial phase of the life cycle of an enterprise, it is not often clear what is the extent of changes that are necessary, and often it is even less clear what is the list of enterprise entities involved in that change. There are many ways in which an enterprise can conduct business. To carry out a change process successfully, the involved enterprises / enterprise entities need to be identified, and their relations must be understood. These relations will

determine the business functions, and they must be chosen carefully to ensure continued operation.
The presented Virtual Enterprise Methodology (VEM) outlines activities to consider when setting up and managing VEs. As a methodology, the VEM helps companies to ask the right questions when preparing for- and setting up an *Enterprise Network*, which works as a breeding ground for setting up VEs. The VEM applies the Virtual Enterprise Reference Architecture (VERA) as an underlying structure. VERA[2] is a specialisation of GERA[3], which is a component of the GERAM Framework.

Chapter 8: Analysing the Present Situation and Refining Strategy (Uppington et al)
Chapter 8 places emphasis on various types of preliminary analyses deemed necessary to support enterprise change processes. Many of these change processes are strategic in nature and thus there is a need for 'strategic input' on behalf of management with leadership abilities. Often, the various strategic alternatives need to be analysed to establish which one is realistically attainable organisationally, technically and financially. The analytic processes described herein are not a replacement for strategy making but are indispensable for supporting strategy making, so as to ensure that the developed strategy is feasible.

Chapter 9: Developing the Enterprise Concept – The Business Plan (Molina)
Chapter 9 provides fundamentals to develop an Enterprise Concept. It provides a framework, methodology and tools to better understand the process of designing an Enterprise. Throughout the Chapter best practices are identified and an easy to follow methodology is presented to guide individuals responsible for planning, designing and creating an enterprise. It provides a set of recommendations for the creation of a competitive enterprise concept including the definition of the mission and vision, creation of strategies, use of performance measures, identification and evaluation of core process and competencies, and finally the preparation of action plans/business plan.

Chapter 10: Enterprise Modelling – the Readiness of the Organisation (Hysom)
Chapter 10 presents an evaluation method through which management can assess enterprise modelling maturity. One of the difficulties in understanding enterprises is that the user is usually working with at least three different

[2] Both VEM and VERA were developed as part of GLOBEMEN (Global Engineering and Manufacturing in Enterprise Networks) - an international IMS (Intelligent Manufacturing Systems, http://www.ims.org) project. Refer to the Technical Research Centre of Finland (VTT)'s web site (http://globemen.vtt.fi/) for details.
[3] Acronym for Generalised Enterprise Reference Architecture

perspectives: reality, perceptions and expectations. Misunderstandings about which one of these are addressed by enterprise models that are developed in an enterprise integration effort can lead to confusion and the feasibility of the change is compromised. In addition, within each of these perspectives there are a number of dimensions which enterprise modelling technology must be able to explore to make good decisions or to perform analysis. These dimensions include structural, behavioural, and value properties. Given the experience, expertise and tools available at any one time to an enterprise, there are limitations to what an enterprise can achieve using enterprise modelling in its present state of maturity.

Chapters 11, 12 and 13: Requirements Specification

Chapter 11: Modelling Functional Information and Resource Requirements (Bernus);
Chapter 12: Modelling the Management System – Enterprise Management Activities and Processes (Olegario et al);
Chapter 13: Modelling Resource Requirements (Zelm)

Chapters 11, 12 and 13 present the objectives and overview the methods for capturing the requirements specification of an enterprise entity. Due to the complexity of enterprise models, they are developed by focusing on different aspects, such as function, information, resource and organization views.
Chapter 11 on function and information modelling is about the methods to determine the mission fulfilment functions and how these functions interact with one another via information and material flow. The Chapter starts with methods to determine what functions the stakeholders would like to include in the specified entity and how these user requirements can be translated into a requirements specification. It continues with the selection of modelling languages for the modelling tasks considering the necessary level of details to be captured in these specifications.
Chapter 12 treats the requirements specification of the management system of enterprise entities. Although Chapters 11 and 13 have dealt with the requirements for functional, information and resource views of the enterprise, the requirements specification of the management and control system calls for special treatment. This Chapter also presents a detailed example of management roles in enterprise networks and virtual enterprises.
Chapter 13 on resource modelling discusses how requirements can be captured to ensure that the required capabilities and capacities of human and technical resources match the needs of the business processes. This Chapter uses the CIMOSA resource modelling constructs for demonstration.
While organisational requirements must be addressed by a requirements specification, these are included in the functional and resource views as required capabilities – attached to functional or resource requirements.

Chapter 14: Enterprise Modelling (Gruninger)
Chapter 14 gives an overview of the state of the art and development directions of the formal basis of enterprise modelling. One of the main drivers of enterprise modelling today is the need for interoperable information systems that are able to link many organisations (such as in electronic business). The role of an enterprise model is to achieve model-driven enterprise design, analysis, control and evaluation. The development and adoption of formal ontological theories (called 'generic enterprise models' in GERAM) that underlie interoperation is indispensable for tool developers and is an important task for international standardisation.

Chapter 15: Preliminary Design : Translating Requirements to Design Specifications (Chen et al)
Chapter 15 addresses preliminary, or 'architectural' design that transforms requirements to design specifications. After identifying the requirements for a future state of the enterprise, e.g. a manufacturing enterprise, the crucial step is to aggregate functions (decisional and manufacturing alike) into functional modules and to assign necessary resources to their daily operations. There are many alternative solutions from which to choose from. How to do this and what tools to use, what rules to follow are the main topics of the Chapter. This Chapter focuses on presenting the concepts and principles, rather than a methodology; however, some of the necessary principles are discussed here.

Chapter 16: Organisational Design (Bernus)
Chapter 16 addresses the problem of assigning roles to individuals in the enterprise. The organisation is described as a relation between tasks & responsibilities and human resources (individuals and groups of individuals). Choices in organisational design have strong influence on the health and functionality of the enterprise, and these choices are presented in this Chapter.

Chapter 17: Application Reference Models and Building Blocks for Management and Control (Rosemann)
In Chapter 17 reference models are characterised as reusable (often semi-formal) descriptions of specified domains. They can be differentiated into industry-, project- and application-specific models. This Chapter focuses on application-specific reference models and discusses one of the most comprehensive models (the SAP reference model) as an example. It will be explained how the underlying modelling techniques could be extended, so as to enable the model to explicitly capture alternative application scenarios.

Chapter 18: Designing the Information Technology Subsystem for Enterprise Integration (Camarinha-Matos et al.)
Chapter 18 deals with the information technology subsystem as a determinant player in the efficiency of enterprises. The main challenge here lies on 'systems integration', as it is mandatory to cope with the large legacy of soft-

ware tools. That task is, however, typically handled in ad-hoc manners, and customised for each enterprise. The need for systems integration, its facets, levels and obstacles are first discussed. Then, a set of general steps for systems integration is suggested. Second, the suggested integration methodology approaches this problem at various levels of complexity, namely cell / shop-floor integration, intra-enterprise integration, and inter-enterprise integration. This Chapter addresses the main applied technologies, typical solutions, main trends, and open challenges for every level of complexity.

Chapters 19-22: Case Studies
Chapter 19: Ford Motor Company's Investment Efficiency Initiative (Nevins et al);
Chapter 20: The Business Process Revolution: Transformation to Process Organization (Levi)
Chapter 21: Farley Remote Operations Support System (Mo)
Chapter 22: The Use of GERAM to Support SME Development in Mexico (Molina et al)

Chapter 19 presents a case study of the Ford investment efficiency initiative. This Chapter, released for re-publication by the Institute for Defence Analyses, was prepared as a case study on integrated product / process development implementation. The work was originally commissioned with the intention of being used for acquisition and technology training purposes in the US Department of Defense. It describes how the Ford Motor Company implements a management-driven initiative to steer cost tradeoffs and cost targeting very early in the product/ process development process and throughout product realisation. The name of this initiative is Investment Efficiency – the ability to simultaneously minimize investment by Ford and to optimise value for the customer.
Chapter 20 describes a large US-based energy generation and trading organization applying process modelling to transform the company into a 'Process Enterprise'. The key technology used in this endeavour is an advanced process-modelling tool that allows the continuous maintenance of enterprise processes and knowledge sharing among all involved stakeholders.
Chapter 21 is a case study of a medium sized enterprise that introduced remote operations support to its world-wide customers for the use, maintenance and troubleshooting of their CNC machines. The Chapter describes how a suitable language was selected to represent process knowledge in remote operations support and how the resulting Internet-based information system provided a number of benefits to the company, including the capture and management of process knowledge.
Chapter 22 is a case study describing how enterprise architecture has been used to support the design, development and operations of SMEs in Mexico. Results are described using the case of a medium sized plastic products company.

The editors are fully aware that there are many other successful applications of enterprise engineering and would like to encourage the readers to send their success stories to them for further editions of this Handbook.
Finally, we would like to express out gratitude to many people who have helped us to put this handbook together. The authors of various Chapters have rewritten their manuscript in order to provide consistence in style and content. Their contributions are highly valued. The editors feel privileged to have been working with such outstanding professionals. Dr Müller and his staff at Springer Verlag have helped us through the long and tedious process of preparing the manuscript. Without the hard work of Mr. Ovidiu Noran, who converted various text formats and figures into the LaTeX, we probably would not have been able to publish such impressive looking volume. Ovidiu has also done fine contribution by inserting the final cross-references and bibliographies in the Chapters and removed unnecessary repetitions, and also provided great help with the final review and editing work. Ms Teresa Squarci has improved the style and done an excellent job by proofreading the manuscript. We are extremely grateful to both of them.

Associate Professor Peter Bernus
Dr Laszlo Nemes FTSE FIEAust
Professor Günter Schmidt

Advisory Board Members of this Handbook

Contents

Part III Defining the Requirements for Enterprise Change

1

INTRODUCTION

Peter Bernus[1] and Laszlo Nemes[2]

[1] Griffith University (Australia)
[2] CSIRO *- Manufacturing and Infrastructure Technology (Australia)

1.1 Enterprise Architecture

With the globalisation of economies enterprises are operating as large, complex networks of autonomous units. Design offices, manufacturing facilities, services and maintenance stations are scattered around the world. Component supplier networks provide vital subassemblies and parts. Such intricate field of endeavour needs a clear operational structure in order to design and manage the continuous metamorphosis necessary not just to keep businesses alive, but also to strive for growing businesses and operate them successfully during their whole of life.

Enterprises are not just activities offering products or services but they can be regarded as 'products' themselves, something that needs to be invented, defined (specified), designed and built, just as any other product – even if architecting enterprises should rely on special methods that reflect the peculiar characteristics of enterprises. This has been clear and obvious for new, 'green field' companies, as they are designed, built, and operated according to the specifications of their stakeholders. This 'product view' is less clearly understood for already operating enterprises. Business objectives, manufacturing practices, organisational structures, and information infrastructures keep evolving trying to provide the most effective framework for agile enterprises. Due to constant change in the environment and in business strategies and objectives enterprises should often be re-designed, and these modifications could be significant. Business priorities have to be analysed, required changes specified, overall structures and details designed and implementation carried out. These sorts of re-designs result renewed enterprises that may be regarded as 'new products'.

To design enterprises and manage them through their entire life, tools and methodologies are needed which are based on fundamental principles. During much of the twentiest century, enterprises have grown organically and their

* Commonwealth Scientific and Industrial Research Organisation, www.csiro.au

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

design has been based on intuition, rather than theory. It is not obvious that a theory of such complex socio-technical systems can be developed and successfully applied, in all enterprising endeavour, and indeed present body of knowledge about enterprise architecture must be assessed and applied in a social and technological context. During the past ten years, however, new and powerful theories have been developed that promise generic applicability, even though enterprises that intend to apply this knowledge need to be at a certain level of maturity.

Engineers have been using analysis and design tools for building complex[3] machines and electrical devices for a long time. They learn these skills and their underlining theories and disciplines at universities. Having mastered this knowledge, graduates are well prepared to design new products. The emerging science of enterprise engineering, or enterprise architecture, similarly includes a system of tools and methods necessary to design and re-design a business enterprise.

There are many types of businesses – such as manufacturing, tourism, health care, government services, banking, defence, insurance or farming, just to name a few. The tenet of enterprise architecture is that all of these should be designed professionally and analysed as necessary, like any other product, so as to successfully realise them. The theories, tools and methodologies are the same for existing and newly created enterprises – irrespective of the nature of the business. Enterprise architecture activities are also the same - whether performed to initially create an enterprise or to subsequently modify it. Creating a 'new' product is after all just a specific case of 'modification', not the least because any 'new' design is necessarily influenced by the knowledge of previous successful (and unsuccessful) designs – someone who has no previous knowledge of business is unlikely to be able to design a new one. Changing an existing product from 'A' to 'B' is clearly an example of development or renewal. However, often the development or renewal process throws away a substantial part of the original product, and in the limit there is almost nothing shared between the old and the new. If the product does not initially exist, then a non-existent product is 'modified' into an existent one. Whichever is the case the design process is always influenced by previous knowledge of similar businesses or components thereof. So, designing a green-field enterprise is a special case of a renewal process. Such simple abstraction has helped the community of enterprise architects to develop a generic approach to treat all cases in a similar way.

[3] Complexity can be measured using various metrics, expressed as some function of the number of elements and relations in a system. These functions grow very rapidly, so that doubling the number of relations in a system, for example, might increase complexity even by an order of magnitude. Above certain levels of complexity, analytic models are hard to develop and even harder to solve, therefore enterprise architects need to apply special methods to reduce the complexity of any *view* of the system that needs to be considered for designing and operating / managing an enterprise.

There are many supporting disciplines in the tool-bag of an enterprise architect, and these tools are shared between enterprises of a great diversity – ranging from a manufacturing business to an expedition to the Mount Everest! Enterprise Architecture Frameworks organise and systematise this bag of tools, and include reference architectures, methodologies and methods (procedures), tools and other components as well, as will be discussed in this book. Importantly, this systematisation must include the definition of common concepts that can be uniformly understood by all involved stakeholders, who belong to different disciplines or have different backgrounds. It is expected that eventually the rudiments of enterprise architecture will be taught in schools in the same way as physics or chemistry, with an ever more detailed and specialised curriculum would be offered in secondary and tertiary education.

Readers who are looking for practical guidance in the field of enterprise architecture may think that a clear definition of these components may be too pedantic or too academic. It is easy to point out, that there are many aspects of daily life where most people have already met such categories – perhaps not even making conscious differentiation between them. The following example will illustrate the functional value of these concepts. It will also show that there may be different experts creating the components but for enterprise architects to use them (who would typically have management or engineering background) they would have to develop familiarity with all of these concepts.

Take the analogy from natural language. Various kinds of reference books are on the shelves to describe the essence of a language. Large encyclopaedias (e.g. Encyclopaedia Britannica) are the results of the efforts of many hundreds of scholars from different backgrounds. They provide detailed information on human knowledge on the particular language. Dictionaries (such as the Oxford English Dictionary) are compiled to provide exact definitions of words and they offer examples on their correct use. A Thesaurus is a type of dictionary in which words with similar meanings are grouped together. All of the above focus on the meanings of words in different contexts, *i.e.*on *semantics.*

Descriptive grammars and books on usage and patterns describe the rules by which words can be combined into sentences. The collective set of rules is called *syntax.* Through knowing the meanings of words and understanding the rules of how to link them together the structure of a particular language can be grasped. This structure may be called the *architecture* of the language. Scholars describe this architecture as a foundation for correct speaking and writing. The word architecture is used here in two different (albeit related) meanings. One can say that the *architecture* of the language is its structure and the characteristics of this structure. On the other hand, the *architecture* of a language also means the activities involved in creating the many descriptions of the language, as well as the relations between these activities.

Teachers have different objectives. They use the above body of knowledge in a form best suited to teach students of that particular language. They use these books as references for themselves and develop different *methodologies*

or methods for various purposes. There are methods for teaching native children in the primary school, and other methods to teach pupils in the secondary school, and the curriculum in tertiary institutions depends on whether the students learn arts or science. Technical universities focus on teaching the correct use of language for writing reports, communicating ideas and giving powerful presentations. All these cases require a different teaching methodology. For example, language students from foreign countries with mother tongues different from what they are learning at school require a special teaching approach. They require teachers to be specialised for teaching foreigners a second language, and the necessary skills and methods of these teachers also depend on how old the students are.
Once a methodology is defined according to the needs of the task at hand these in turn specify the prerequisite skills and knowledge that teachers and student must have. As a result, it is possible to identify what kinds of *tools* will be needed: such as books, audio tapes, video tapes interactive CDs, pictures, etc. The relation between architecture, methodology and tools is depicted in Table 1.1.

Table 1.1. Relation between elements of. Architecture, Methodology and Tools

Architecture	Teaching Methodology for	Tools
Encyclopaedias	Native speakers	Books
Dictionaries	Second language	Videos
Descriptive grammars	Adults	Interactive media
Thesaurus	Children	Classroom tutorials
Usage & patterns	Professional areas	Language laboratories

This analogy can be used for discussing the life-cycle of an enterprise, for example a manufacturing enterprise. The architecture defines the structure of the enterprise through its life-cycle, which starts with the identification of the enterprise, followed by conceptual design including strategy, mission, vision and values. This overall outline ('concept') is necessary to be able to complete a preliminary (or 'architectural') design, which in turn is necessary to be able to complete the detailed design of the enterprise. These decisions (the identification of the business concept, the strategy and the rest of the 'concept', as well as the preliminary design) may be captured in descriptions, or models, and can be used by a detailed design and implementation phase to actually establish the enterprise. In the case of a factory one will see here the building erected, machines purchased and installed, hardware and software installed, people trained or hired, the organisation established, and all external services secured. When everything is completed the factory if ready for operation. The final phase in the life-cycle is the decommissioning of the manufacturing enterprise, when it reaches the end of its useful life.

The above life-cycle activities may be repeated many times, when during the life of the enterprise, as part of continuous development or renewal. Enterprise Architecture Frameworks usually represent these life-cycle activities and their relationships on a diagram, called Enterprise Reference Architecture. For example, the Purdue Enterprise Reference Architecture (PERA) visualises these life-cycle phases as shown in Fig. 1.1 (a detailed explanation of this model will be given in Chapter 3).

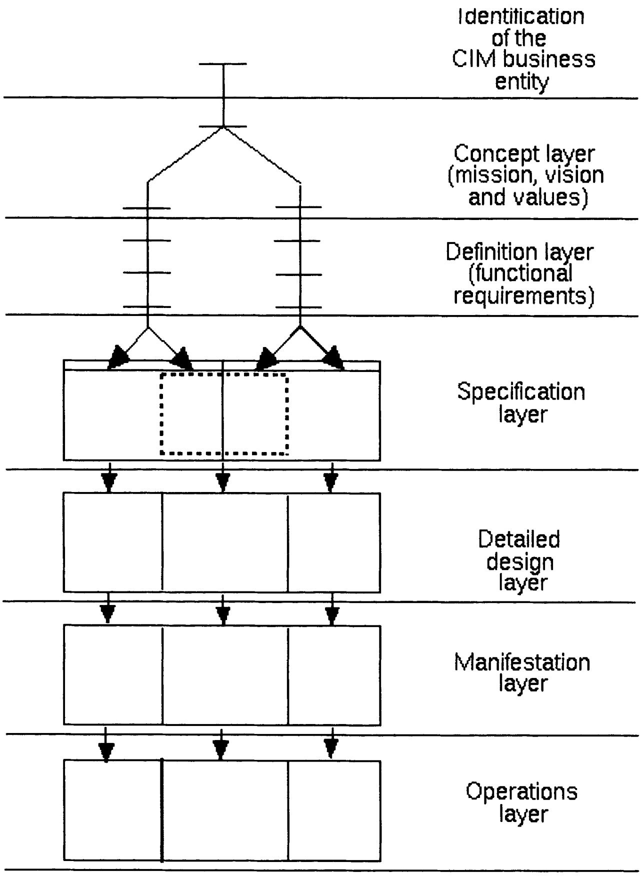


Fig. 1.1. The Purdue Enterprise Reference Architecture (PERA) shows life-cycle activities in 'layers' (or 'phases')

A Reference Architecture shows the anatomy of the life-cycle of an enterprise. It describes the logical structure of activities. Like in all functional diagrams,

the interfaces between functions are shown by arrows representing information or data exchange.
The *methodology* for going through the various activities in the life-cycle of an enterprise can be described and communicated in a generic way – such as has been done, for example in the Purdue Guide for Master Planning (Williams et al. , 1996). Furthermore, the methodology may be supported by software tools, which provide a form of workflow management to guide enterprise architects through the steps of an enterprise architecture project. There are some 20~30 major methods already supported with software tools. Unfortunately, an uninitiated user of these methodologies and associated tools can easily be mislead, because not every change process requires all methods and all tools, or even all steps. A generic methodology must be *tailored* for the specific use, so as to include all relevant steps, and only relevant steps, to select modelling languages and tools, as well as to identify any reusable components that can speed up development time. While the method of tailoring is still an art, and needs experienced professionals, it is also an actively researched problem with the aim of developing future tools that will be able to help the end user to generate a tailored methodology for any specific case of enterprise architecture endeavour.
One of the most important tools of enterprise architecture are Enterprise Modelling Tools that support the use of various Enterprise Modelling Languages. Many modelling languages are available for tool developer – some defined as international standards, and some accepted as de-facto standards. The Structured Analysis and Design Methodology (SADT - (Ross , 1977)) – which was a combination of enterprise modelling languages and methods to use them – was developed in the early seventies. This was later adopted and extended to become the IDEF family of languages (with the most notable components being IDEF0, IFE1X and IDEF3). A number of software vendors use these languages as the basis their for their modelling and simulation tools. These and many other tools will be discussed in Chapter 3 and the languages in Chapter 14. Table 1.2 illustrates the Architecture-Methodology-Tools triplet for enterprises.

Table 1.2. Necessary components of Enterprise Architecture Frameworks

Architecture	Methodology for	Tools
Enterprise Reference Architecture	Master planning and implementation 	Modelling languages Modelling Tools Reusable Models Reusable Building Blocks

Once the enterprise is up and running it is in its *operational* phase. After some time, new requirements will emerge. This is caused either by external business requirements or by internal recognitions that the applied practice should be updated or modified. Whatever the reasons are, management will initiate a *renewal* project. A team is usually formed to identify the needs for change and a specification is put together for the renewed enterprise. This project is itself behaving as an enterprise and has to be managed accordingly. For example, the same life-cycle phases can be observed for project enterprises as described earlier. There is no limit to this nesting of enterprise life-cycles and the view of such cascading models proves extremely useful. As this 'project enterprise' operates, it covers some or all life-cycle activities (phases) of the renewed enterprise, and the recursive view can clearly define the mission of this project enterprise (refer Figure 1.2). This view has been successfully implemented in the through life support of major infrastructure objects.

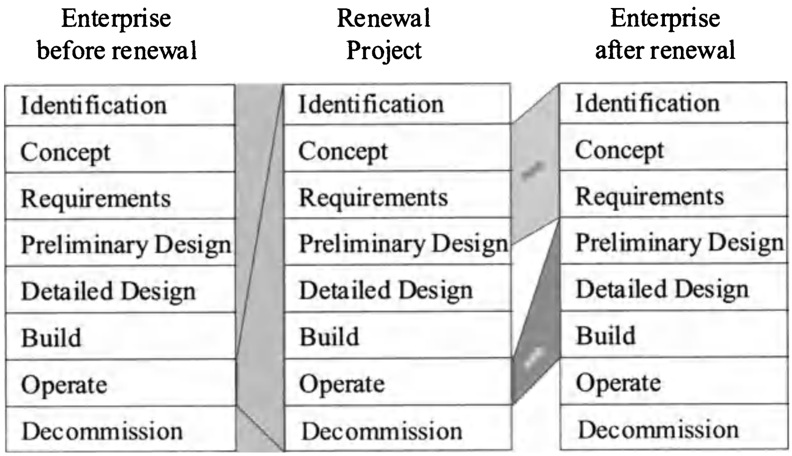


Fig. 1.2. The relation between an enterprise and its renewal project

Many engineering contractors and defence contractors have realised that selling products to customers makes up less than one third of the business value that they could generate if the undertook the through life support or their products , extending the business to any after sales service, such as maintenance, operational support, renewal, refurbishment or upgrade.
It is important to understand that the architectural view describes the anatomy of such enterprises. It tells everything about the structure but noth-

ing about *when* the activity will be performed – time is deliberately abstracted from in the architecture (or life-cycle) view of enterprise entities. Furthermore, there is no obstacle for the same type of life-cycle activity to be performed at the same time, e.g. it is quite conceivable that a number of 'renewal' projects would be carried out at the same time. At a given time, different projects would typically be in different stages (performing life-cycle activities pertinent to that stage) as not all projects start at the same time and certainly not all of them are of the same size. This view is consistent with modern project management practice, and is based upon the fact that project enterprises display similar structures in their life-cycle view.

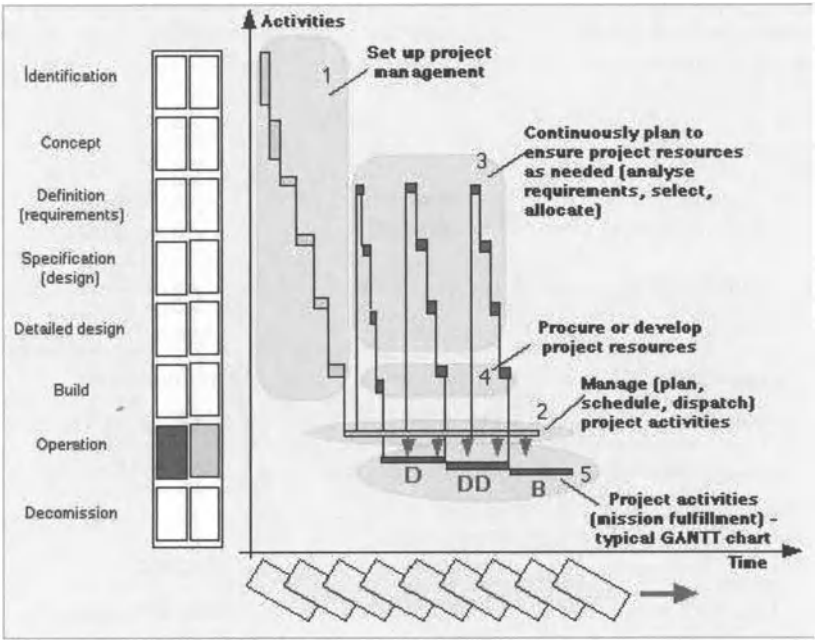


Fig. 1.3. Life cycle and the timeline: an example of activities in a project enterprise

Readers might be still a little bit confused with the concept of architecture, as they do not yet see the time aspect on this diagram. It is therefore useful to explain the relation between the life-cycle architecture diagram and the timeline where life-cycle activities unfold (refer Fig. 1.3). The architecture diagram is drawn symbolically on the left hand side of the figure in a form resembling a 'chocolate bar'. On the right side of the diagram the life-cycle activities are represented as they are carried out in time. As it is clear from this figure some activities are carried out in sequential order, and some overlap. There could be loops and repetitions due to iterations between say design and build activities,

e.g. when these actions are done in parallel – especially when the project is a large-scale installation. The relationship between the two diagrams is strong, but it is not one to one. The timeline (the 'life history')gives more detail on *how* various phases are managed, therefore it gives a realistic, physical view of the process. This life-history may be subdivided into stages, with milestones in between, similar to traditional project management practice. The architecture diagram represents the relationships between phases, and shows the information exchange between adjacent life-cycle activities, as well as draws the attention to the fact that while a project might be seen as very complex, in reality it consists of a small set of activity types that need to be managed.
The core business of an enterprise includes the development and delivery of products and services, the development of processes and practices that perform the above activity, as well as the development and delivery of management processes for the enterprise to operate as expected. For this reason a very important part of enterprise architecture is the development of business process definitions, as discussed in the following Section. Since these processes are interconnected by an information and material flow, the information structures necessary to establish these connections (the *integration* of processes) must also be developed (sometimes called 'information architecture'). Finally, processes are performed by people and machines, therefore the resources and their allocation to processes – according to the processes' desired parameters – need to be determined as well. Since processes (both enterprise engineering and production & service processes) are performed in time and in locations, these descriptions need to be situated in time and space.
Over and above the variety of descriptions according to their use in the enterprise architecture process, it must be noted that the backgrounds, motivations and positions of people who are involved in enterprise architecture are very different as therefore the presentation of these descriptions must fulfil the needs of these people. It is thus of little wonder, that the tools needed for designing enterprises are of such a considerable variety and complexity.

1.2 The Business Process Perspective

The traditional concept of computer integrated manufacturing aimed at providing computer assistance for various elements of product development, manufacturing and management processes. Integration here meant to 'link' one computer aided activity to the other to provide a continuous information flow among them.
Enterprise integration goes a big step beyond that on two accounts: a) it asks questions about the core business processes in an organisation and concentrates on how to perform these tasks better b) the information and material flow in the entire process is considered, .whether it involves machines (manufacturing tools, computers, communication devices) or people.

In this context, Enterprise Architecture is practiced to achieve Enterprise Integration, through capturing and describing processes, strategies, organisational structures, resources, goals and constraints of the enterprise. It specifies the business process requirements, identifies the solution options, presents alternative designs and provides implementation paths at strategic, tactical and operational levels[4].
The entire business process is studied and described in terms of three interlinking networks.

- Material flow and transformation (this process services customer needs.)
- Information flow and transformation (knowledge and information processing of any form)
- Supporting organisational structure of people who actually carry out the non-automated parts of the tasks in the two above-mentioned processes. This component is essential, since people make decisions based on experience and incomplete information.

In order to manage the business processes effectively, enterprise integration methodologies are needed which rely on life-cycle architectures, and use modelling techniques or languages, and implementation know-how.
It is possible to position the concept of business process re-engineering (BPR) in the practice of enterprise architecture. From the above discussion, it should be clear that the redesign ('re-engineering') of business processes is part of a larger set of methods, and depending on the circumstances and the objectives of the change process might take on the form of a radical redesign of the processes as used in the enterprise's present practice (according to the original tenet of the BPR movement this is always so), or take on a continuous improvement approach, where present processes are analysed, expressed in business process models, and improved as necessary. Thus, the tools of BPR are part of the toolset of enterprise architecture, while its methods might be appropriate in one case and inappropriate in another – depending on the objectives of the change.

1.3 Extended, Virtual Enterprises

The traditional view of an enterprise is that of a hierarchical organisation. In large companies, with diversified business profiles, an enterprise may mean a branch office, a division, or a plant that operates as a relatively independent business unit.
Very few companies, however, design and manufacture every component of their products in-house. In fact, all agile enterprises co-operate with a large

[4] Note that in the business community the name 'tactical level' refers to a longer term activity then the 'operational level', while in defence terminology the name 'operational level' refers to a longer term activity then the 'tactical level'.

number of suppliers and sub contractors, in most life-cycle phases of their products, E.g. components and subassemblies are manufactured outside the premises of the product vendor, detailed design tasks might be subcontracted, and after sales service may be provided by a third party. More recently, this extended enterprise is being designed in close cooperation with partners in a supply network, and some suppliers are in strategic relations with a number of product vendors. For example, the relation between a network, a one-of-a-kind project enterprise and its product may be as represented in Fig. 1.4, where the assumption of this particular example is that the product is identified and specified by the network.

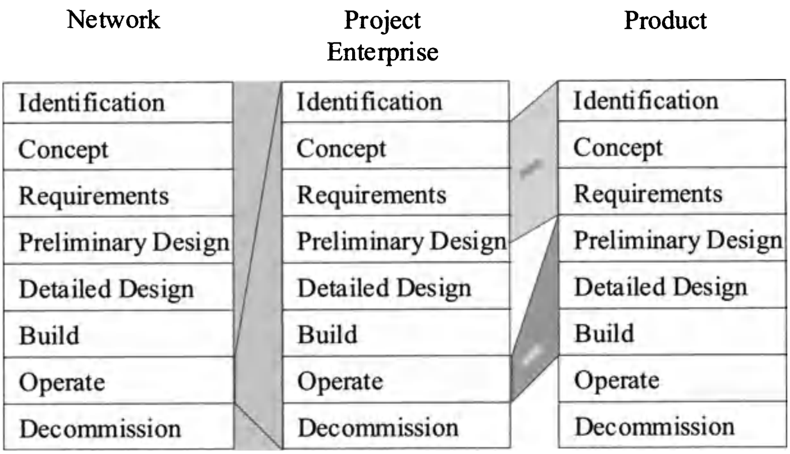


Fig. 1.4. An example relation between a Network, a Project Enterprise and its Product

When considering the business processes of an enterprise, the scope of consideration must include all value-adding activities throughout the supply chain, not only the activities that are performed within a given company. For example, a company A could have a business processes of which a part is being performed outside the physical boundaries of the company, e.g. by company B. In many cases, company B's core business requires the contribution of yet another company ('C') through C delivering some components or subassemblies, etc. Enterprise engineering tools therefore should be able to cope with the highly distributed and co-operative nature of the extended enterprise con-

cept, and if this co-operation is based on a highly dynamic allocation of roles then with the concept of virtual enterprise (Goldman et al. , 1995).
The business decision 'make or buy' (or 'make or have it made') is certainly a key factor when defining an extended integrated enterprise. Business activities (and business focus) may change quickly and accordingly so should business process. The dynamism, the time dimension of enterprise design and implementation through the whole life-cycle, is critical and will be even more critical in the future. Virtual or extended enterprises will be formed and dissolved quickly and the underlying business processes will change as well.

1.4 Enterprise Integration Methods

Integrated manufacturing systems in the eighties and nineties were individually specified and designed. Typically, it took 1-1.5 years to summarise the customer's requirements and to develop a comprehensive set of specifications. These specifications were never complete and contained undetected contradictions. Because of the lack of formal descriptions, it was impossible to model the expected functions. Furthermore, when the implementation was complete, it took a long time to test and modify these systems. These custom built integrated systems resulted in monolithic enterprises. Once they were installed, it was extremely difficult to update and to further develop them. As design methodologies became more sophisticated, the design phase became shorter, but still all systems were individually made and crafted. It became obvious that integrated enterprises can not be fully tailor-made, but should be created like any other industrial product. Furthermore, enterprises should be designed for their whole life cycle – meaning that the design should factor in both the operational requirements and the requirements for modifiability, maintainability, as well as the requirement that parts of the system would have to be decommissioned in the future.
Previous ESPRIT projects, funded through the European Union, have developed integration architectures, modelling tools, and demonstrated their use in industrial applications. The most important results have been the Computer Integrated Manufacturing - Open System Architecture (CIMOSA) project, its supporting modelling tools and its validation in industrial environments (CIMOSA Association , 1994). These results are restricted to various industrial consortia in Europe, but they are accessible to researchers through the CIMOSA Association. The GRAI/LAP laboratory of the University of Bordeaux has developed a framework and modelling tools to support enterprise integration (Doumeingts et al. , 1992). They are keen to prove their results in practice and have many industrial references. The Purdue consortium has developed an engineering oriented architecture (Williams , 1992) and an associated implementation methodology (Williams et al. , 1996). They were published as the Purdue Enterprise Reference Architecture and its implementation manual. Toronto University is developing a complete set of enterprise

modelling tools to support any such architecture. The Zachman Framework (Zachman , 1987) was developed for organising the many models that information systems development requires, and can be mapped to the above mentioned architectures. The C4ISR (Department of Defence Architecture Framework, or DoDAF (C4ISR Architecture Working Group , 1997)) was developed to achieve similar aims in the military sector. The ARIS framework was developed for organising models of information systems throughout their development (Scheer , 1992).

It is probably less well known that the presently widely used term of Business Process Re-Engineering, promoted by consultants and understood by company executives is just one, albeit important aspect of Enterprise Integration (EI). BPR, as it is abbreviated, focuses on the review and re-design of the core business process of an enterprise and it uses either common sense practice or professional tools offered by the EI community. There is a close cooperation between researchers of BPR and EI. What BPR does not address at this stage is the need to continuously engineer the business process, instead of re-engineering it once.

Concurrent Engineering (CE) traditionally has been viewed as a powerful method to reduce time to market. It incorporates both technological and human-organisational issues. Concurrent design activities will span the entire supply chain with the product vendor and subassembly (component) suppliers simultaneously working on the design of the new product. Therefore, Concurrent Engineering is one facet of EI: CE is a collection of methods and tools used to carry out cooperation between distributed engineering activities.

1.5 The Generalised Enterprise Reference Architecture And Methodology (The GERAM Enterprise Architecture Framework)

All these previously mentioned architectures and integration methods – as developed in the 1970s and 1980s – have produced many useful results, but they influenced one another very little. There was a need to compare and evaluate them and to combine the various methodologies and modelling techniques and to identify any missing elements. Based on the result of the investigations by an international Task Force on Enterprise Integration, established in 1992 by the International Federation of Information Processing (IFIP) and the International Federation of Automatic Control (IFAC), the authors proposed a new Generalised Enterprise Reference Architecture and Methodology (GERAM (Bernus and Nemes , 1994)[5]). This proposal was then fully developed by this Task Force and became the basis of the International Standard ISO 15740:2000. GERAM defines a tool box of concepts for designing and

[5] See also the Final Report of the IFIP/IFAC Task Force on Enterprise Integration at http:///www.cit.gu.edu/au/~bernus/taskforce [2002]

maintaining enterprises during their entire life-cycle. GERAM is therefore about those methods, models and tools which arc needed to build integrated enterprises. The architecture is also generic because it applies to most, potentially all types of enterprises. It is notable, that all the above Architecture Frameworks can be mapped onto GERAM, which is very useful, because users of these individual frameworks can benefit from identifying what their specific architecture framework offers and where additional tools or methods might be necessary. In fact, during the development of GERAM several contributing frameworks have undergone such extension.

Chapter 2 of the Handbook contains the definition of GERAM, while Chapter 3 contains a description of the CIMOSA, PERA, GRAI/GIM. Zachman and C4ISR(DoDAF) frameworks and of their mapping onto GERAM.

1.5.1 Requirements for a Generalised Enterprise Reference Architecture and Methodology

1. The architecture should cover all activities necessary for designing, operating, maintaining and improving enterprises. Since architectures are the backbone of the complex methodology necessary to engineer the complex business processes, the architecture should cover the entire life cycle
2. There is a need for a consistent modelling environment leading to executable code. The ideal modelling environment should be modular so that alternative methodologies could be incorporated into it. There should not be any restrictions built into the modelling languages as to what methodologies they would be applied for. The modelling environment should be extensible (rather than be a closed set of models) and permissive, leaving space for alternative modelling methods. (CIMOSA and TOVE are good examples).
3. Comprehensive and easy to follow methodology must be technically correct and easy to use by multidisciplinary teams, in designing enterprises within budget, expected time frame and resource constraints. The enterprise integration methodology should also be open and expandable. It should be presented both in generic and special forms. The former ones contain the general rules and the latter ones contain more detailed rules and serve particular industry sectors. (GIM methodology and PERA have many industrial applications).
4. The cost and the time frame of the enterprise engineering process should be kept low. Adoption of reusable, tested (and standard) models (building blocks) greatly help this process. Enterprise integration and engineering are complex processes and are carried out by groups of people. This should be supported by sound design theory and collaborative technology based on multiple agents.
5. The GERAM should provide provision for unifying products, processes, management, enterprise development and strategic management. Architectures should tie and relate enterprise integration and enterprise en-

gineering to the rest of the activities in the enterprise. It is especially important to support evolutionary path (migration) to integrated enterprises.

1.5.2 The Structure of the Generic Enterprise Reference Architecture and Methodology

To satisfy the above-mentioned requirements, the components of the GERAM Framework are the following:

1. Generic Enterprise Reference Architecture (**GERA**). An architecture is a description (model) that collects (and systematises) all concepts necessary to describe the structure, the elements and their relationships as well as all relevant properties (part, functions, cost, time, resources, etc.) of a device, a system or an enterprise. GERA is the definition of the enterprise related concepts, with primary focus on the life cycle of the enterprise.
1. Generic Enterprise Engineering Methodology (**GEEM**). This is a generic description of the process of enterprise integration. The methodology is a detailed model of this process with instructions on each step of the integration project.
3. Generic Enterprise Modelling Tools and Languages (**GEMT&L**). Engineering of the integrated enterprise is a highly sophisticated (multidisciplinary management, design, and implementation) exercise, during which various forms of descriptions and models of the enterprise need to be created. More than one modelling language is needed to describe an enterprise.

1.5.2.1 Tested Components To Build Enterprise Models

1. Generic Enterprise Models (**GEMs**). Generic Enterprise Models define the concepts which are used in enterprise modelling. These definitions may be captured in an informal way (e.g. in a glossary), but there is a growing need to define them in a formal manner, in so-called Ontological theories. Ontological theories (OTs) describe (define) enterprise related modelling concepts. Ontological theories capture, for example, the meaning of concepts like function, activity, process, event, resource, time, structure, dynamics, costs, etc. The reason why such formal definitions are necessary is because modelling tools should be able to interpret enterprise models, check them for correctness and completeness, etc. The rich set of concepts used in enterprise modelling can also be presented as so-called *meta-models.*
2. Partial Enterprise Models (**PEMs**). The development of enterprise models is a complex and time-consuming task. To accelerate this activity, the architect must draw on previous experience, which may be captured in earlier created models and reused in new projects. Complete database

schemata, process definitions, systems of management may be documented in this way, and these designs can be used as building blocks in the process of enterprise integration. The collective name for such reusable models is 'Partial Models' or 'Reference Models'. GERAM defines several typical forms of reference models (refer Chapter 2)

3. Generic Enterprise Modules (**GMs**). These are products or standard implementations of components that are likely to be used in enterprise integration, either by an enterprise integration project itself or by the enterprise. Generic modules (software or hardware systems) can be configured to form more complex modules for the use of individual enterprises. Products form product families, where components can be freely configured to build a complete system. However, today's challenge is to integrate components that originate from different product families, giving rise to a new family of integration infrastructure components. These latter are designed so as to be able to build 'systems of systems'.

1.5.3 Enterprise Integration Clearinghouse

This initiative intends to collect all information on the major projects on Enterprise Integration. It updates its databases on enterprise modelling languages, meta-models, and associated tools. It reports on developments and validations of the major Enterprise Reference Architectures and other Tools and Infrastructures for Enterprise Integration. It contains information on major events related to Enterprise Integration and the mailing list of the people interested in the topics. The Clearing House in Australia is maintained by Griffith University and it is freely accessible on the World Wide Web.

1.5.4 Conclusion

The GERAM architecture was presented to the IFIP/IFAC Task Force on Enterprise Integration. Among others, the international committee invited experts from the CIMOSA Association, the Purdue Consortium and GRAI/LAP Laboratory to discuss and evaluate the proposal. The committee endorsed GERAM as a baseline document for future developments and has mapped the existing architectures to GERAM. However, as architecture frameworks develop these mappings need to be refreshed from time to time. Chapter 3 presents such a mapping of the most popular architecture frameworks.
GERAM in its matrix form (as it was originally developed by the authors) is a powerful research tool. Because of its complex graphical representation it was difficult to be used directly by consultants in enterprise architecture projects and a more visual representation of the architecture was needed to fully exploit its practical value and as part of the Task Force's activities. T J Williams proposed a three dimensional representation to explain GERAM. However, it is expected that the matrix form will be used by researchers for developments and the three dimensional model will be preferred by educators to introduce GERAM into engineering and management practice.

References

Bernus,P., Nemes,L. (1994) A Framework to Define a Generic Enterprise Reference Architecture and Methodology. Invited session ICARV'94, Singapore, (November 1994) pp88-92. also in *Computer Integrated Manufacturing Systems* 9,3 (July 1996) pp 179-191

C4ISR Architecture Working Group (AWG) (1997) C4ISR Architecture Framework, V 2.0. US Department of Defense, December 1997.

CIMOSA Association (1994) CIMOSA, Open System Architecture for CIM , October 1994.

Doumenigts, G., Vallespir, B., Zanettin, M. and Chen, D. (1992) GIM, a GRAI integrated methodology. A method for designing CIM systems. Report. GRAI/LAP, University of Bordeaux, France. May, 1992.

Goldman,S.L., Nagel, R.N. and Preiss, K. (1995) Agile Competitors and Virtual Organisations. Van Nostrand Reihold Publishing Inc.

Ross, D. (1977) Structures Analysis (SA): A Language for Communicating Ideas, TSE 3 (1), pp16-34

Scheer, A.W. (1992) Architecture of Integrated Information Systems, Berlin : Springer.

Williams, T.J. (1992) The Purdue Enterprise Reference Architecture. Instrument Society of America. Research Triangle Park, North Carolina, USA. 1992.

Williams,T.J., Rathwell, G.A. and Li, H. (Eds) (1996) A Handbook on Master Planning and Implementation for Enterprise Integration Programs, Report #160. Purdue Laboratory for Applied Industrial Control, West Lafayette: Purdue University. June 1996

Zachman, J.A. (1997) A Framework for Information Systems Architecture. *IBM Systems Journal*, 26(3).

Part I

Architecture Frameworks - Organising Enterprise Architecture Knowledge

2

GERAM: THE GENERALISED ENTERPRISE REFERENCE ARCHITECTURE AND METHODOLOGY

Version 1.6.3 (Final*)

IFIP–IFAC Task Force on Architectures for Enterprise Integration **

* This is a minor revision of GERAM v1.6.3 – with some editorial changes and explanations that do not affect the technical content. GERAM V1.6.3 also appears as an Annex A to ISO15704:2000.

** The work of the IFIP Task Force on Architectures for Enterprise Integration (1990-2002) was supported by a large number of members. Special thanks should go to Jim Nevins (USA) who originally proposed the establishment of this TF. Prof T.J.Williams (Purdue University) has lead the TF with great energy from the TF's inception till 1996, and his efforts and leadership were invaluable to this group. The list of active members who attended the numerous meetings is long, and it is hard to make sure that there are no omissions made when we commend the efforts of Kurt Kosanke (CIMOSA Association), Jim Nell (NIST), Guy Doumeingts and David Chen (Bordeaux U), Francois Vernadat (U Metz), Richard Weston (U Loughborough), David Shorter (ITFocus), Yoshiro Fukuda (Hosei U), Laszlo Nemes (CSIRO), Jacob Vlietstra (formerly AT&T), Gus Olling (Daimler Chrysler), Michael J Rodd (UC Swansea), Ljubo Vlacic (Griffith U), Johan Vesterager (TU Denmark), Michael Petit (U Namour), Hong Li (Waterlogic), Atsui Inamoto (Mitsubishi E), Ziquiong Deng (Narvik Inst T), Zenjin Han (Tsinghua U), Yuilu Chen (Tshinghua U), Arturo Molina (U Monterrey), Bruce Gutheridge (Systemhouse), Michael Gruninger (U Toronto/NIST), Luis Basanez (UP Catalunya), Marco Garetti (U Milan), Mark Fox (U Toronto), Pinto Fereira (U Porto), John Edwards (Loughborough U), Jochem Roland (IPK Berlin), Christopher Bussler (Oracle), Jim Brown (UC Galway), Gunter Schmidt (U Saarlandes), Sam Bansal (SAP), Cheng Leong Ang (Gintic), Brane Kalpic (ETI), Yusaku Shibata (Nagoya U), James Brosvic (Honeywell), Bruno Vallespir (Bordeaux U) and Dick Zoetekouw (AT&T), F Gh Filip (ICI Inst.), Geza Haidegger and Gyorgy Kovacs (SzTAKI), Peter Kopacek (TU Wien), Angel Ortiz (U Valencia), Ajit Pardasani (NRC Canada), Norbert Rozenich (BMf Wien), . We also thank other members of the Task Force who supported the work but have not been able to participate the meetings: S.K. Banerjee (U Strathclyde), V Charpulat (EERIE), F Fadel, Joel Favrel (AIPRAO Lyon), M Jarke (GMD), HJ Molstad (Standards Denmark), J Mylopoulos (U Toronto), Mihai Avram (IPA-SA), Leo Obrst (MITRE)

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

2.1 Introduction

2.1.1 Background

This chapter presents GERAM, which is the result of a ten year project by the IFIP-IFAC Task Force generalising the contributions of a number of Enterprise Architecture Frameworks. As a generalisation, GERAM allows different Frameworks to be assessed for completeness, as well as be further developed. The GERAM Framework includes a number of useful concepts in Enterprise Architecture which were earlier not precisely defined and this lack of definition prevented practitioners and researchers to exchange their results. GERAM is the basis of several international standards, most notably ISO 15704:2000 and ISO/DIS 19439. In addition, GERAM can also be used as an Enterprise Architecture Framework itself.

GERAM includes a life-cycle reference architecture (GERA), which via its modelling framework defines the scope of enterprise modelling. The definition of Enterprise Entities and their recursive relationships allows various Business Models to be explored and designed, and the differentiation in GERAM between the concepts of *life-cycle* and *life history* helps users to relate management views of enterprises with project management.

One of the most important characteristics of today's enterprises is that they are facing a rapidly changing environment and can no longer make predictable long term provisions. To adapt to this change enterprises themselves need to evolve and be reactive so that change and adaptation would be a natural dynamic state rather then something occasionally forced onto the enterprise. This necessitates the integration of the enterprise operation[2] and the development of a discipline that organises all knowledge that is needed to identify the need for change in enterprises and to carry out that change expediently and professionally. This discipline is called Enterprise Engineering[3].

Previous research work, carried out by AMICE Consortium on CIMOSA (CIMOSA, 1996), by GRAI Laboratory on GRAI and GIM (Doumeingts et al. , 1998), and by the Purdue Consortium on PERA (Williams , 1994b), (as well as similar methodologies by others) has produced reference architectures which were meant to be organising all enterprise integration knowledge and serve as a guide in enterprise integration programs. The IFIP/IFAC Task Force analysed these architectures and concluded that even if there were some overlaps, none of the existing reference architectures subsumed the others; each of them had something unique to offer. The recognition of the need to define a generalised architecture is the outcome of the work of the Task Force.

[2] Enterprise Integration is about breaking down organisational barriers and improving interoperability to create synergy within the enterprise to operate more efficiently and adaptively.

[3] Enterprise Engineering is the collection of those tools and methods which one can use to design and continually maintain an integrated state of the enterprise.

Starting from the evaluation of existing enterprise integration architectures (CIMOSA, GRAI/GIM and PERA), the IFAC/IFIP Task Force on Architectures for Enterprise Integration has developed an overall definition of a generalised architecture. The proposed framework was entitled 'GERAM' (Generalised Enterprise Reference Architecture and Methodology). GERAM is about those methods, models and tools, which are needed to build and maintain the integrated enterprise, be it a part of an enterprise, a single enterprise or a network of enterprises (virtual enterprise or extended enterprise).
GERAM, as presented below, defines a tool-kit of concepts for designing and maintaining enterprises for their entire life-history. GERAM is not yet-another-proposal for an enterprise reference architecture, but is meant *to organise existing enterprise integration knowledge.* The framework has the potential for application to all types of enterprise. Previously published reference architectures can keep their own identity, while identifying through GERAM their overlaps and complementing benefits compared to others.

2.1.2 Scope

The scope of GERAM encompasses all knowledge needed for enterprise engineering / integration. Thus GERAM is defined through a pragmatic approach providing a generalised framework for describing the components needed in all types of enterprise engineering / enterprise integration processes, such as:

- Major enterprise engineering/enterprise integration efforts (green field installation, complete re-engineering, merger, reorganisation, formation of virtual enterprise or consortium, value chain or supply chain integration, etc.);
- Incremental changes of various sorts for continuous improvement and adaptation.

GERAM is intended to facilitate the unification of methods of several disciplines used in the change process, such as methods of industrial engineering, management science, control engineering, communication and information technology, i.e. to allow their combined use, as opposed to segregated application.
One aspect of the GERAM framework is that it unifies the two distinct approaches of enterprise integration, those based on product models and those based on business process design. It also offers new insights into the project management of enterprise integration and the relationship of integration with other strategic activities in an enterprise.
An important aspect of enterprise engineering is the recognition and identification of feedback loops on various levels of enterprise performance as they relate to its products, mission and meaning. To achieve such feedback with respect to both the internal and the external environment, performance indicators and evaluation criteria of the corresponding impact of change on

process and organisation are required. The continuous use of these feedback loops will be the prerequisite for the continuous improvement process of the enterprise operation and its adaptation to the changes in the relevant market, technology and society.

2.2 The Framework for Enterprise Engineering and Enterprise Integration

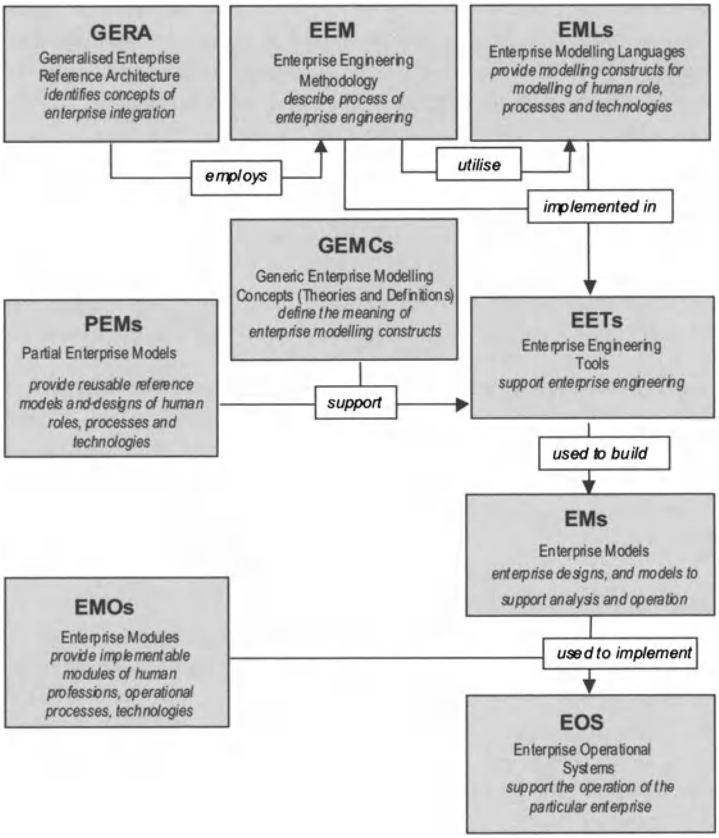


Fig. 2.1. GERAM[4]framework components

GERAM provides a description of all the elements recommended in enterprise engineering and integration and thereby sets the standard for the collection

[4] Generalised Enterprise Reference Architecture and Methodology

of tools and methods from which any enterprise would benefit to successfully tackle initial integration design, and the change processes which may occur during the enterprise operational lifetime. It does not impose any particular set of tools or methods, but defines the criteria to be satisfied by any set of selected tools and methods. GERAM views enterprise models as an essential component of enterprise engineering and integration; this includes various formal (and less formal) forms of design descriptions utilised in the course of design – as described in enterprise engineering methodologies – such as computer models, and text and graphics based design representations.

The set of components identified in GERAM is shown in Fig. 2.1 and is briefly described in the following. Each component is then defined in detail in Section 2.3.

The GERAM framework identifies in its most important component GERA (Generalised Enterprise Reference Architecture) the basic concepts to be used in enterprise engineering and integration (for example, enterprise entities, life-cycles and life histories of enterprise entities). GERAM distinguishes between the methodologies for enterprise engineering (EEMs) and the modelling languages (EMLs) that are used by the methodologies to describe and model, the structure, content and behaviour of the enterprise entities in question. These languages will enable the modelling of the human part in the enterprise operation as well as the parts of business processes and their supporting technologies. The modelling process produces enterprise models (EMs) that represent all or part of the enterprise operations, including its manufacturing or service tasks, its organisation and management, and its control and information systems. These models can be used to guide the implementation of the operational system of the enterprise (EOSs) as well as to improve the ability of the enterprise to evaluate operational or organisational alternatives (for example, by simulation), and thereby enhance its current and future performance.

The methodology and the languages used for enterprise modelling are supported by enterprise engineering tools (EETs). The semantics of the modelling languages may be defined by ontologies, meta models and glossaries that are collectively called generic enterprise modelling concepts (GEMCs). The modelling process is enhanced by using partial models (PEMs), which are reusable models of human roles, processes and technologies.

The operational use of enterprise models is supported by specific modules (EMOs) that provide prefabricated products like human skill profiles for specific professions, common business procedures (e.g. banking and tax rules) or IT infrastructure services, or any other product which can be used as a component in the implementation of the operational system (EOSs).

Potentially, all proposed reference architectures and methodologies could be characterised in GERAM so that developers of particular architectures could gain from being able to commonly refer to the capabilities of their architectures, without having to rewrite their documents to comply with GERAM. Users of these architectures would benefit from GERAM because the GERAM

definitions would allow them to identify what they could (and what they could not) expect from any chosen particular architecture in connection with an enterprise integration methodology and its proposed supporting components.

2.2.1 Definition of GERAM Framework Components

2.2.1.1 GERA - Generic Enterprise Reference Architecture

GERA defines the enterprise related generic concepts recommended for use in enterprise engineering and integration projects. These concepts can be categorised as:

a) Human oriented concepts
   1) to describe the role of humans as an integral part of the organisation and operation of an enterprise and
   2) to support humans during enterprise design, construction and change.
b) Process oriented concepts for the description of the business processes of the enterprise;
c) Technology oriented concepts for the description of the business process supporting technology involved in both enterprise operation and enterprise engineering efforts (modelling and model use support).

2.2.1.2 EEMs - Enterprise Engineering Methodology

EEMs describe the processes of enterprise engineering and integration. An enterprise engineering methodology may be expressed in the form of a process model or structured procedure with detailed instructions for each enterprise engineering and integration activity.

2.2.1.3 EMLs - Enterprise Modelling Languages

EMLs define the generic modelling constructs for enterprise modelling adapted to the needs of people creating and using enterprise models. In particular enterprise modelling languages will provide construct to describe and model human roles, operational processes and their functional contents as well as the supporting information, office and production technologies.

2.2.1.4 GEMCs - Generic Enterprise Modelling Concepts

GEMCs define and formalise the most generic concepts of enterprise modelling. Generic Enterprise modelling concepts may be defined in various ways. In increasing order of formality generic enterprise modelling concepts may be defined as:

- Natural language explanation of the meaning of modelling concepts (glossaries);
- Some form of meta model (e.g. entity relationship meta schema) describing the relationship among modelling concepts available in enterprise modelling languages;
- Ontological Theories defining the meaning (semantics) of enterprise modelling languages, to improve the analytic capability of engineering tools, and through them the usefulness of enterprise models. Typically, these theories would be built inside the engineering tools.

2.2.1.5 PEMs - Partial Enterprise Models

PEMs (reusable-, paradigmatic-, typical models) - capture characteristics common to many enterprises within or across one or more industrial sectors. Thereby these models capitalise on previous knowledge by allowing model libraries to be developed and reused in a 'plug-and-play' manner rather than developing the models from scratch. Partial models make the modelling process more efficient

The scope of these models extends to all possible components of the enterprise such as models of humans roles (skills and competencies of humans in enterprise operation and management), operational processes (functionality and behaviour) and technology components (service or manufacturing oriented), infrastructure components (information technology, energy, services, etc.).

Partial model may cover the whole or a part of a typical enterprise. They may concern various enterprise entities such as products, projects, companies, and may represent these from various points of view such as data models, process models, organisation models, to name a few.

Partial enterprise models are also referred to in the literature as 'Reference Models', or 'Type I Reference Architectures'[5]. These terms have the same meaning.

2.2.1.6 EETs - Enterprise Engineering Tools

EETs support the processes of enterprise engineering and integration by implementing an enterprise engineering methodology and supporting modelling languages. Engineering tools should provide for analysis, design and use of enterprise models.

2.2.1.7 EMs - (Particular) Enterprise Models

EMs represent the particular enterprise. Enterprise models can be expressed using enterprise modelling languages. EMs include various designs, models

[5] Life-cycle architectures such as GERA are referred to as 'Reference Architectures of type II'

prepared for analysis, executable models to support the operation of the enterprise, etc. They may consist of several models describing various aspects (or views) of the enterprise.

2.2.1.8 EMOs - Enterprise Modules

EMOs are products that can be utilised in the implementation of the enterprise. Examples of enterprise modules are human resources with given skill profiles (specific professions), types of manufacturing resources, common business equipment or IT infrastructure (software and hardware) intended to support the operational use of enterprise models.
Special emphasis is on the IT infrastructure which will support enterprise operations as well as enterprise engineering. The services of the IT infrastructure will provide two main functions:

a) model portability and interoperability by providing an integrating infrastructure across heterogeneous enterprise environments;
b) model driven operational support (decision support and operation monitoring and control) by providing real-time access to the enterprise environment.

The latter functionality will be especially helpful in the engineering tasks of model update and modification. Access to real world data provides much more realistic scenarios than for model validation and verification than simulation based on 'artificial' data.

2.2.1.9 EOSs - (Particular) Enterprise Operational Systems

EOSs support the operation of a particular enterprise. Their implementation is guided by the particular enterprise model which provides the system specifications and identifies the enterprise modules used in the implementation of the particular enterprise system.

2.3 Description of GERAM Framework Components

2.3.1 GERA – Generalised Enterprise Reference Architecture

2.3.1.1 General

GERA defines the generic concepts recommended for use in enterprise engineering and integration projects. These concepts can be classified as:

a) Human oriented concepts: They cover human aspects such as capabilities, skills, know-how and competencies as well as roles of humans in the enterprise organisation and operation. The organisation related aspects have

to do with decision level, responsibilities and authorities, the operational ones relate to the capabilities and qualities of humans as enterprise resource elements. In addition, the communication aspects of humans have to be recognised to cover interoperation with other humans and with technology elements when realising enterprise operations.

Modelling constructs will be required to facilitate the description of human roles as an integral part of the organisation and operation of an enterprise. The constructs should facilitate the capture of enterprise models that describe:

1) human roles,
2) the way in which human roles are organised so that they interoperate with other human and technology elements when realising enterprise operations and
3) the capabilities and qualities of humans as enterprise resource elements.

An appropriate methodology will also be required that promote the retention and reuse of models that encapsulate knowledge (i.e. know-how possessed by humans expressed as an enterprise asset) during the various life phases of enterprise engineering projects.

b) Process oriented concepts: They deal with enterprise operations (functionality and behaviour) and cover enterprise entity life-cycle and activities in various life-cycle phases; life history, enterprise entity types, enterprise modelling with integrated model representation and model views;
c) Technology oriented concepts: They deal with various infrastructures used to support processes and include for instance resource models (information technology, manufacturing technology, office automation and others), facility layout models, information system models, communication system models and logistics models.

Examples of enterprise reference architectures are provided by ARIS[6], PERA[7], CIMOSA[8], GRAI/GIM[9], IEM[10]. ENV 40003 defines a general Framework for Enterprise Modelling. ISO DIS 14258 defines Rules and Guidelines for Enterprise Models.

2.3.1.2 Human Oriented Concepts

The role of humans in the enterprise remains fundamental. However sophisticated and integrated an enterprise can be, humans will always make the final decision. With the emergence of decentralised organisations, flat hierarchies,

---

[6] ARIS: Architektur fur Informations Systeme (Architecture for Information Systems)

[7] PERA: Purdue Enterprise Reference Architecture

[8] CIMOSA: CIM Open Systems Architecture

[9] GRAI/GIM: Graphe Résultats et Activitées Interreliées (Graphs with Results and Activities Interrelated)/ GRAI Integrated Methodology

[10] IEM: Integrated Enterprise Modelling

responsibility and authority delegation, the knowledge about the roles of individuals and who is responsible for what becomes an invaluable asset for any enterprise especially those operating according to new management paradigms. Therefore, capturing this knowledge in enterprise models will prove to be very useful and enable flexible reaction to environmental changes. In addition the different factors describing the capabilities of humans have to be captured as well. Human factors are concerned with professional skills, experience, etc.

Typically, humans may assume different roles during enterprise engineering and operation. Examples are: chief executive, chairperson, marketing, sales, technical (R&D), finance, engineering and manufacturing directors, product design, production planning, information systems, quality, product support, logistics, capital equipment, shop floor and site managers; assistant managers, accountants, cashiers, product, process and information system designers, production engineers, electrical and mechanical technicians, maintenance personnel, quality inspectors, supervisors and foremen, machine operators, storeroom and inventory persons, progress chasers, secretaries, drivers, cleaners, management and systems consultants, systems integrators, system builders, and IT suppliers and vendors.

Often humans and groupings of humans will be assigned a number of roles and responsibilities that need to be carried out concurrently and cohesively, where each may involve different reporting lines and control procedures. Furthermore their roles can be expected to change over time as process requirements change and individual and group capabilities advance or decline. The ability to manage and deploy human resources effectively and collectively under complex and changing circumstances is key to the competitive position of an enterprise.

Although it is not practical to model all aspects of human roles within an enterprise, concepts are required to formally represent those human factors connected with enterprise integration. This should be achieved in a way that harmonises human roles with that of other human and technology elements, as an integral part of the organisation and operation of an enterprise. Hence the need for constructs that promote the capture of knowledge (possessed by humans) in the form of reusable enterprise models about:

- the role of individuals and groups of individuals,
- the way in which organisational structures and constraints are applied to co-ordinate those roles, such as via the delegation of responsibilities and control and reporting procedures, and
- the capabilities and qualities of humans, treated as resource elements.

It is important to understand when, by whom and how decisions are made in the enterprise as well as who can fulfil certain tasks in the replacement of others.

Knowledge about the roles of humans and ways in which those roles can be harmonised can be capitalised and reused as an enterprise asset. The degree to which such knowledge can be formalised within computer processable models

will directly influence the degree to which it can be capitalised. Computer processable models naturally facilitate analysis, transformation, storage and integration (based on common understandings). Whereas mental models retained and processed by humans will be less tractable for such purposes. However the retention and reuse of informal models (such as in the form of cause and effect relationships and shared mental models or images) can also be of significant benefit in realising improved cohesion in an enterprise. Hence even where formal modelling of human issues proves impractical the retention and reuse of knowledge should be encouraged by deploying suitable social processes, human organisational structures, methodologies and tools that promote explicit model capture and visualisation.

The ability to retain and reuse human factors knowledge can be of vital importance to the competitive position of an enterprise. Reuse of human knowledge can enable an enterprise to:

- respond rapidly to new market opportunities or changes in environmental conditions;
- re-engineer its business (and manufacturing) processes
- improve its management and utilisation of resources as new products and services are launched;
- improve its resilience to the loss of core competencies in the form of knowledgeable human assets.

2.3.1.2.1 Human Role Models

A taxonomy of human factors and their relation to the activity model would allow to relate human aspects to enterprise models. Needed are human role models on decision making, capabilities, socio-technical models (motivation, incentives, ...), skill models, organisational models, with others to be determined.

Human role models will support the definition of human responsibilities and authorisation in both the enterprise operation and its organisation. Such models will support the collection of relevant information and its recognition in the design of the operational system. GERA caters for human factors in its view concept (see 2.3.1.5.2). This concept provides in its process oriented model views and technology oriented implementation view for the recognition of humans and the capturing of relevant information. Also the role of humans as an operational resource is recognised in these views. In this role the human skills and capabilities will be described. The human aspect of enterprise integration must also be thoroughly dealt with in the change methodology both in the human's role of change agent and in the role of potential and actual resource (see 2.3.2).

2.3.1.3 Process Oriented Concepts

Business process-oriented modelling aims at describing the processes in the enterprise capturing both their functionality (what has to be done) and their

behaviour (when things are done and in which sequence). In order to achieve a complete description of the processes a number of concepts have to be recognised in the guiding methodology. The process-oriented concepts defined in GERA are

- enterprise entity life-cycle and life-cycle phases,
- life history,
- enterprise entity types, and
- enterprise modelling with integrated model representation and model views.

These concepts will be described in more detail in the following sections.

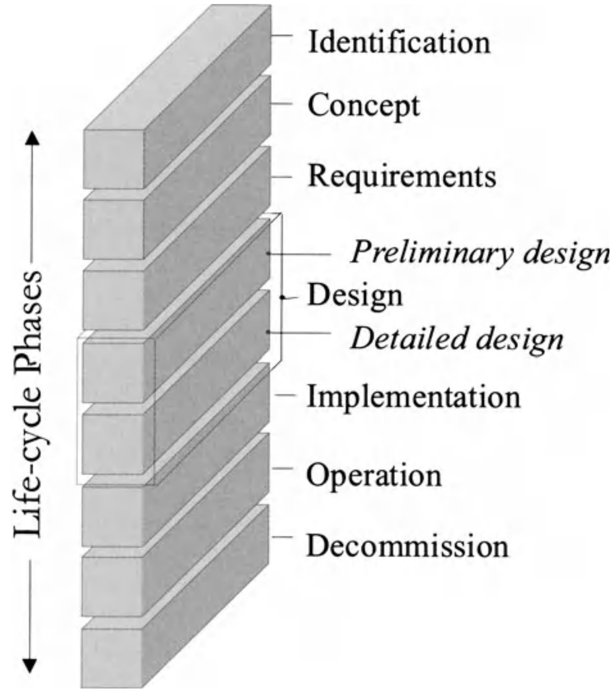


Fig. 2.2. GERA Life-cycle phases for any enterprise or entity[12]

2.3.1.3.1 Life-cycle

Figure 2.2 shows the GERA life-cycle for any enterprise or any of its entities. The different life-cycle phases define types of activities that are pertinent

[12] Phase in GERAM refers to the level of abstraction and detail to which the given entity is developed and does *not* imply succession in time.

during the life of the entity. Life-cycle activities encompass all activities from identification to decommissioning (or end of life) of the enterprise or entity. A total of seven life-cycle activity types have been defined, that may be subdivided further as demonstrated for the design type activities that have been broken down into two lower level types of activities (based on the customary subdivision in many industries of design into preliminary- and detailed design activities). The life-cycle diagram used in the description of the life-cycle of an entity is itself a model of the enterprise engineering methodology.

2.3.1.3.1.1 Entity Identification This is the set of activities that identifies the contents of the particular entity under consideration in terms of its boundaries and its relation to its internal and external environments. These activities include the identification of the existence and nature of a need (or need for change) for the particular entity. In other words these are the activities that define what is the entity of which the life-cycle is being considered.

2.3.1.3.1.2 Entity Concept The set of activities that are needed to develop the *concepts* of the underlying entity. These concepts include the definition of the entity's mission, vision, values, strategies, objectives, operational concepts, policies, business plans and so forth.

2.3.1.3.1.3 Entity Requirement The activities needed to develop descriptions of operational requirements of the enterprise entity, its relevant processes and the collection of all their functional, behavioural, informational and capability needs. This description includes both service and manufacturing requirements, management and control requirements of the entity – no matter whether these will be satisfied by humans (individuals or organisational entities), or machinery (including manufacturing-, information-, control-, communication-, or any other technology).

2.3.1.3.1.4 Entity Design The activities that support the specification of the entity with all of its components that satisfy the entity requirements. The scope of design activities includes the design of all human tasks (tasks of individuals and of organisational entities), and all machine tasks concerned with the entity's customer services and products and the related management and control functions. The design of the operational processes includes the identification of the necessary information and resources (including manufacturing-, information-, communication-, control- or any other technology).
Any life-cycle phase may be subdivided into sub-phases to provide additional structuring of life cycle activities. Example in Fig. 2.2: dividing design into functional design or specification and detailed design to permit the separation of:

a) overall enterprise specifications (sufficient to obtain approximate costs and management approval of the ongoing project), and

b) major design work necessary for the complete system design suitable for fabrication of the final physical system.[13]

2.3.1.3.1.5 Entity Implementation The activities that define all those tasks that must be carried out to build or re-build (i.e. manifest) the entity. This comprises implementation in the broadest sense, covering

a) commissioning, purchasing, (re)configuring or developing all service, manufacturing and control software as well as hardware resources;
b) hiring and training personnel, and developing or changing the human organisation;
c) component testing and validation, system integration, validation and testing, and releasing into operation;

Note that the implementation description (documentation) may deviate from the design specification of the entity due to preferences or unavailability of specified components.

2.3.1.3.1.6 Entity Operation The activities of the entity that are needed during its operation for producing the customers product or service which is its special mission along with all those tasks needed for monitoring, controlling, and evaluating the operation. Thus the resources of the entity are managed and controlled so as to carry out the processes necessary for the entity to fulfil its mission. Deviations from goals and objectives or any feedback from the environment may lead to requests for change, which includes enterprise re-engineering or continuous improvement of its human and technology resources, its business processes, and its organisation.

2.3.1.3.1.7 Entity Decommissioning These activities are needed for disbanding, re-missioning, re-training, redesign, recycling, preservation, transfer, disassembly, or disposal of all or part of the entity at the end of its useful life in operation.

2.3.1.3.2 Life history

The life history of a business entity is the representation in time of tasks carried out on the particular entity during its entire life span. Relating to the life-cycle concept described above, the concept of life history allows to

[13] Note that (a) the need for such subdivision is found methodologically very important (see the Purdue Guide for Master Planning), and (b) the wording allows for the consistency of this life-cycle phase definition with the ENV 40 003 which has only one design phase. The reason for this difference is that the Purdue Guide considers PERA and thus GERA as the model of the methodology, and in that case the subdivision is essential. On the other hand CIMOSA and thus ENV 40 003 considers the life-cycle phases to be characterisations of modelling levels or languages. From this latter aspect the subdivision is not necessarily essential, because the preliminary and detailed design differ only in design detail. Using this wording GERA can play both roles; a and b.

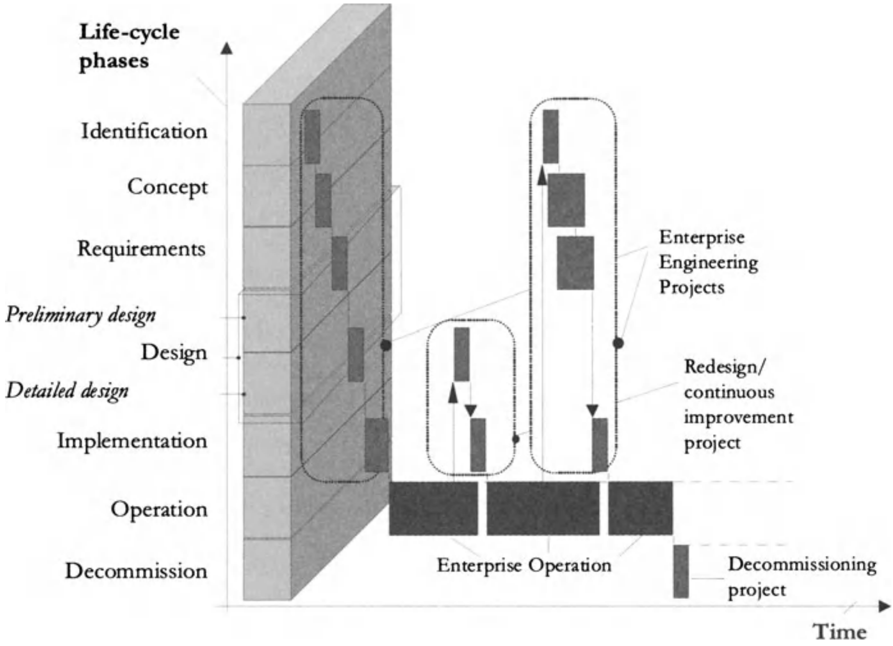


Fig. 2.3. - Parallel processes in the entity's life-history

identify the tasks pertaining to these different phases as activity types. This demonstrates the iterative nature of the life-cycle concept compared with the time sequence of life history. These iterations identify different change processes required on the operational processes and/or the product or customer services.

Typically, multiple change processes are in effect at any one time, and all of these may run parallel with the operation of the entity. Moreover, change processes may interact with one another. Within one process, such as a continuous improvement project, multiple life-cycle activities would be active at any one time. For example, concurrent engineering design and implementation processes may be executed within one enterprise engineering process with considerable time overlap, and typically in parallel with the enterprise operation.

Life histories of entities are all unique, but all histories are made up of processes that in turn rely on the same type of life-cycle activities as defined in the GERA life-cycles (2.3.1.3.1). For this reason life-cycle activities are a useful abstraction in understanding the life history of any entity.

Figure 2.3 illustrates the relations between life-cycle and life history representing a simple case with a total of seven processes: three engineering processes, three operational processes, and one decommissioning process.

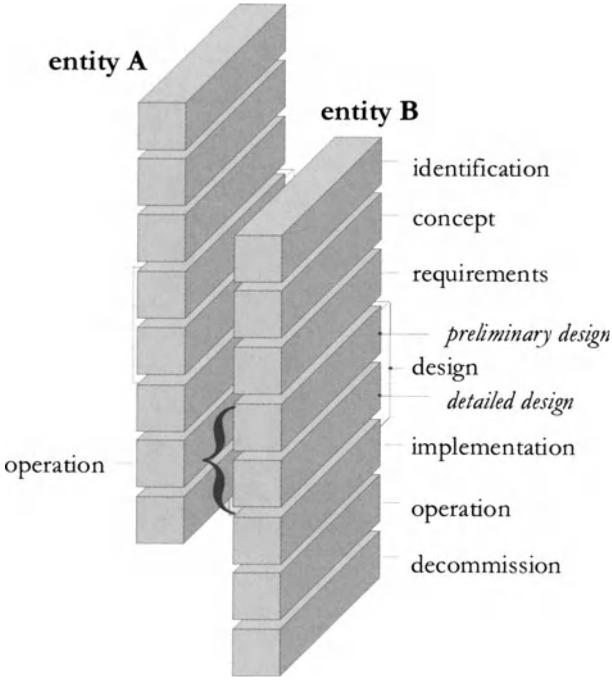


Fig. 2.4. Example for the relationship between life-cycles of two entities

Note: The life history of an entity may be subdivided into *stages*, each stage being an interval in time during which the entity is in a given state. E.g. an entity might be in the specification stage, the development stage, or an operation stage. The activities that are carried out in a given stage are made up of life-cycle activities. E.g. an entity might be in a continuous development *stage*, meaning that the entity is operating, but there are minor detailed design and re-building activities going on at the same time. This difference between the meanings of phase and stage are unfortunately often a source of confusion. While the relation between adjacent phases is not temporal, stages succeed one another in time, often separated by *milestones*. To add to the confusion of terminology projects often name stages after the *main* life-cycle activity that is carried out in the given stage. E.g. when we talk about the identification and concept *stage*, this means that the main activity during this time is the identification and concept development of the entity. However, during this stage there may be a need to carry out specification and design activities as well (even experimental implementation and operation), in case the feasibility of the concept can not be established without such experiments. Therefore while such an identification and concept stage might involve 95% identification and concept phase activities, the remaining 5% of activities may include other life-cycle phase activities as well.

2.3.1.3.3 Entity types in Enterprise Integration

Figure 2.4 shows how the life-cycle activities of two entities may be related to each other. The operation of entity A *supports* the life-cycle activities for design and implementation of entity B. For example, entity A may be an engineering entity producing part of entity B, such as a factory.
Conversely the life-cycle activities of entity A need to be supported with information about the life-cycle details of entity B. That is, to identify a plant, to define its concepts and requirements, and to design it one must use information about which life-cycle activities of the plant's products need to be covered in the operation of this plant.
Examples of other relations between the life-cycle activities of enterprise entities may be defined. However, it is always the case that only the operational activities of entities influence the life-cycle activities of other activities. GERA introduces the concept of entity types and the relations between the different types. Many categories of enterprise entities can be defined. In the following two different ways of categorising enterprise types will be discussed: an operation oriented set and a generic and recursive set of enterprise entity types. The two sets have close relations to each other and both identify the product entity as the result of the operation of other entities.

2.3.1.3.3.1 Operation oriented Enterprise Entity Types These enterprise entity types are all concerned with different types of operations, as follows:

- Project Enterprise Entity (Type A):
  This type defines an enterprise (often with a short life history) that is created for the one-off production of another entity. Examples of project enterprise are: enterprise engineering project, one of a kind manufacturing projects, building projects, etc.
  The project enterprise is characterised by its close linkage with the life-cycle of the single product or service that it is producing. The management system of project enterprises is typically set up quickly, while the rest is created and operated in lock-step with the life-cycle activities of the product of the project.
  Project enterprises are normally associated with, or created by repetitive service and manufacturing enterprises. A typical example would be an engineering project created by an engineering enterprise.
  The products of project enterprises may be diverse, such as large equipment, buildings etc., or an enterprise (e.g. a plant, or an infrastructure enterprise).
- Repetitive Service- and Manufacturing Enterprise Entity (Type B):
  This type defines enterprises supporting a type or a family of products, produced in a repetitive or sustained mode. During their life history these business enterprises undergo multiple change processes. Examples of repetitive business enterprise are service enterprises, manufacturing plants, engineering firms, infrastructure enterprises, etc.

The products of the repetitive service and manufacturing enterprise may be diverse, such as non-enterprise product entities (see below); or products that are enterprises themselves, e.g. project enterprises are regularly created by engineering and building companies.

- Product Entity (Type C):
  This type defines a very large class of entities including any artificial product, such as customer goods, services, hardware equipment, computer software, etc. These entities are not enterprises themselves, but their life-cycles are described by GERAM.

2.3.1.3.3.2 Recursive Enterprise Entity Types A generic and recursive set of four enterprise entity types which have been defined as follows:

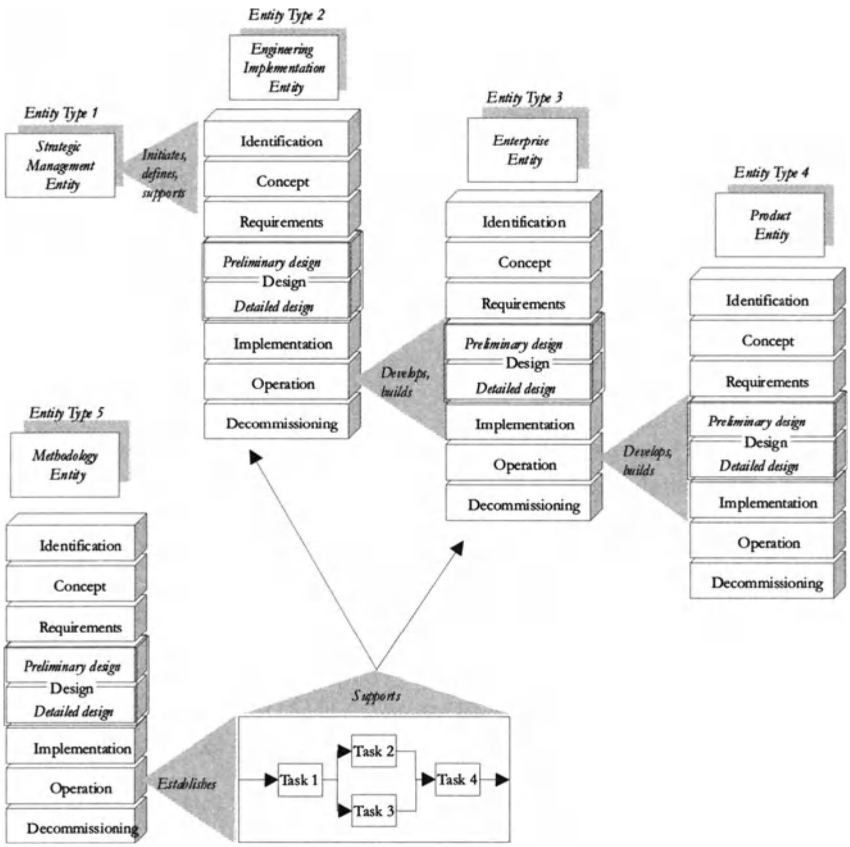


Fig. 2.5. Relationships between life-cycles of GERA entity Types

- Strategic Enterprise Management Entity (Type 1): defines the necessity and the starting of any enterprise engineering / integration effort.
- Enterprise Engineering/Integration Entity (Entity Type 2): provides the means to carry out the enterprise engineering efforts defined by enterprise Entity Type 1. It employs a methodology (Entity Type 5) to define, design, implement and build the operation of the enterprise entity (Entity Type 3).
- Enterprise Entity (Entity Type 3): is the result of the operation of Entity Type 2. It uses a methodology (Entity Type 5) and the operational system provided by Entity Type 2 to define, design, implement and build the products and customer services of the enterprise (Entity Type 4).
- Product Entity (Entity Type 4): is the result of the operation of Entity Type 3. It represents all products and customer services of the enterprise.

This set may be complemented by a fifth entity type that represents the methodology needed for guiding the enterprise engineering and enterprise integration activities.

- Methodology Entity (Entity Type 5): represents the methodology to be employed by any enterprise entity type in the course of its operation, which operation in general leads to the creation of another entity type.

The recursiveness of the first four entity types can be demonstrated by identifying the role of the different entities, their 'products' and the relations between them. Figure 2.5 represents the chain of enterprise entity developments. The Entity Type 1 will always start creation of any lower level entity by identifying goal, scope and objectives for the particular entity. Development and implementation of a new enterprise entity (or new business unit) will then be done by a Entity Type 2 whereas a Entity Type 3 will be responsible for developing and manufacturing a new product (Entity Type 4). With the possible exception of the Entity Type 1 all enterprise entities will have an associated entity life-cycle. However, it is always the operational part of the entity life-cycle in which the lower entity is defined, created, developed and built. The operation itself is supported by an associated methodology for enterprise engineering, enterprise operation, product development and production support.
Figure 2.5 shows both the life-cycle of the methodology (Entity Type 5) and the process model of the methodology. There must be a clear distinction between the life-cycle of the methodology (that is essentially the description of how a methodology is developed) and its process model which is the individual manifestation of the methodology entity itself used to support the operational phase of particular enterprise entities.
The operational relations of the different entity types are also shown in Fig. 2.6 which demonstrates as an example the contributions of the different entities to the life-cycle of a manufacturing entity (Entity Type 3). The manufacturing

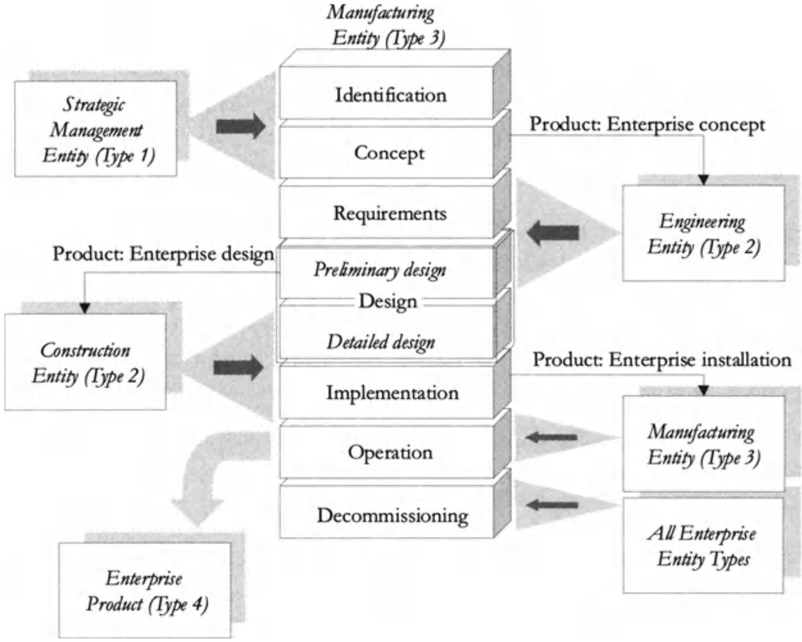


Fig. 2.6. Relationships between GERA entity Types

entity itself produces the enterprise product (Entity Type 4) in the course of its operation phase.

The defined set of entity types is seen to be sufficient to allow representation of other types as well, for example, the distinction between one-of-a-kind or project related enterprise entities and continuous operation type enterprises would only require different parts of the life-cycle activities to be used in the life history of such entities. This is already indicated in Fig. 2.3 in which the engineering processes could relate to an Entity Type 2 and the operational processes to an Entity Type 3 that produces the product or customer services (Entity Type 4). The involvement of Entity Type 1 depends on the degree of change to be carried out in the change process.

2.3.1.3.4 Process Modelling

Process modelling is the activity that results in various models of the management & control as well as the service and production *processes*, and their relationships to the resources, organisation, products etc. of the enterprise. Process modelling allows to represent the operation of enterprise entities and entity types in all their aspects: functional, behaviour, information, resources and organisation. This provides for operational use of the models for decision support by evaluating operational alternatives and for model driven operation control and monitoring.

2.3.1.4 Technology Oriented Concepts

Both the enterprise engineering process and the operational environment employ a significant amount of technology. This is either production oriented and therefore involved in producing the enterprise products and customer services, or management and control oriented – providing the necessary means for communication and information processing and information sharing. Technology oriented concepts have to provide descriptions of the technology involved in both the enterprise operation and the enterprise engineering efforts.
For the operation oriented technology such concepts have to relate to models such as resource models and resource organisation models (e.g. shop floor models, system architectures, information models, infrastructure models), communication models (e.g. network models), etc.
All of these descriptions are applicable in the enterprise engineering environment as well. In addition, there are specific needs for information technology for the support of enterprise engineering (e.g. engineering tools, model development services and model enactment services for animation, simulation, and model based operation control and monitoring).

2.3.1.4.1 IT Support for Enterprise Engineering and Enterprise Integration.

IT support for enterprise engineering as well as enterprise operation should provide two main functions:

a) Model portability and interoperability by providing an integrating infrastructure across heterogeneous enterprise environments.
b) Model driven operational support (decision support and operation monitoring and control) by providing real time access to the enterprise environment.

To enable an integrated real time support of the operation, both the process descriptions and the actual information have to be available in real time for decision support, operation monitoring and control, and model maintenance.

2.3.1.4.2 Enterprise Model Execution and Integration Services (EMEIS)

To illustrate the potential use of computer executable models for on-line operation of the enterprise, Fig. 2.8 illustrates the concept of an integrating infrastructure linking the enterprise model to the real world systems. Integrating services act as a harmonising platform across the heterogeneous system environments (IT and others) and provide the necessary execution support for the model. The process dynamics captured in the enterprise model act as the control flow for model enactment. Therefore access to information and its transfer to and from the location of use is controlled by the model and supported by the integrating infrastructure. The harmonising characteristics of the integrating infrastructure enables transfer of information across and

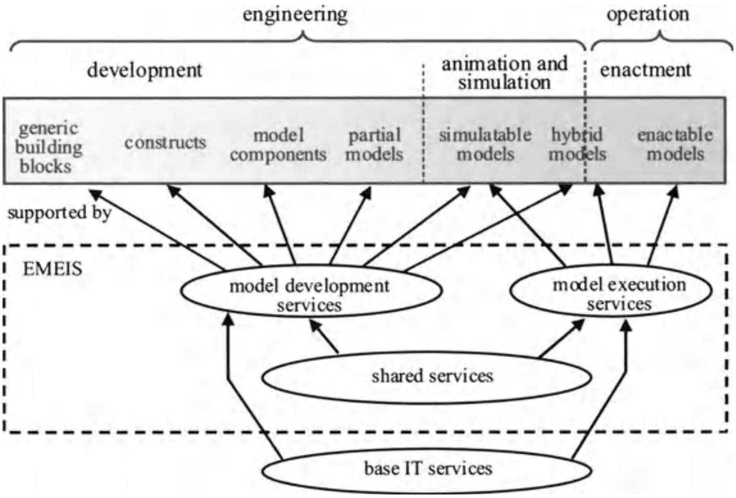


Fig. 2.7. Reference Model of EMEIS

beyond the organisation. Through the semantic unification of the modelling framework interoperability of enterprise models is assured as well.
Efforts aimed at enterprise modelling support have been implemented in pilot implementations (CCE/CNMA, CIM-BIOSIS, CIMOSA, MIDA, OPAL, PISA, TOVE). Some of these project results have been evaluated in a CEN/TC310 report and have lead to statement of requirements for enterprise model execution and integration services by CEN/TC310 as well. The statement of requirements distinguishes between the model development services (MDS), the model execution services (MXS) and the general IT services (see Fig. 2.8). However. no explicit service entities have been defined.
Relevant standardisation is in progress on European level (see *"CIM Systems Architecture - Enterprise Model Execution and Integration Services"* (CEN/TC310 , 1995a,b))

2.3.1.5 Modelling Framework of GERA

GERA provides an analysis and modelling framework that is based on the life-cycle concept and identifies three dimensions for defining the scope and content of enterprise modelling:

[14] The left hand side represents the reference models, while the right hand side represents the resulting particular enterprise models.

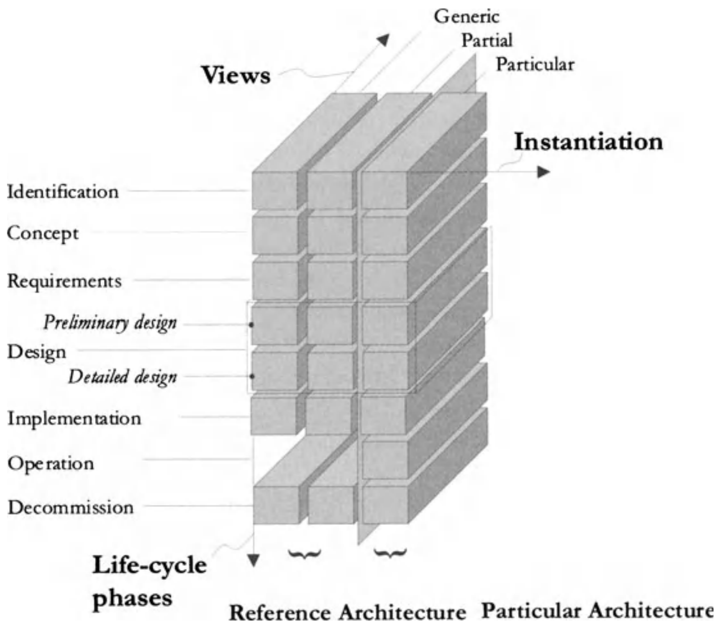


Fig. 2.8. The GERA Modelling Framework[14]

- Life-Cycle Dimension: providing for the controlled modelling process of enterprise entities according to the life-cycle activities.
- Genericity Dimension: providing for the controlled particularisation (instantiation) process from generic and partial to particular.
- View Dimension: providing for the controlled visualisation of specific views of the enterprise entity.

Figure 2.9 shows the three dimensional structure identified above that represents this modelling framework. The reference part of the modelling framework itself consists of the generic and the partial levels only. These two levels organised into a structure the definitions of concepts, basic and macro level constructs (the modelling languages), defined and utilised for the description of the given area . The particular level represents the results of the modelling process - which is the model or description of the enterprise entity at the state of the modelling process corresponding to the particular set of life-cycle activities. However, it is intended that the modelling languages should support the two-way relationship between models of adjacent life-cycle phases. That is, the derivation of models from an upper to a lower state or the abstraction of lower models to an upper state, rather than having to create different models for the different sets of life-cycle activities. (Note: the GERA modelling framework has become the basis of the European and International Standard EN/IS 19439 : 2002. Framework for Enterprise Modelling.)

2.3.1.5.1 Enterprise Modelling

Enterprise modelling is the activity that results in partial or particular *enterprise models* (e.g. various models of the management and control as well as service and production processes, resources, organisation, products etc. of the enterprise). The life-cycle activities of an entity may define various models of that entity to be created. That is, the results of enterprise modelling are all the various designs, models prepared for analysis, executable models to support enterprise operation, and so on (Bernus et al. , 1998).

The emphasis in enterprise modelling is currently on process and product models for representing enterprise operations. Process oriented modelling allows to represent the operation of enterprise entities and entity types in all their aspects: functional, behaviour, information, resources and organisation. This provides for operational use of the models for decision support by evaluating operational alternatives and for model driven operation control and monitoring.

Enterprise models in general represent a very complex reality. In order to reduce this complexity enterprise models have to allow the representation of certain aspects (views) of the model. Aspects that represent part of the model that is relevant to the concerns of the user. This allows the manipulation of the model according to the user's concerns, without being disturbed by the overall complexity of an overall total model.

Enterprise modelling is not limited to process modelling of the enterprise. All other customary design and analysis activities that create descriptions, or models of the enterprise in any phase of the life-cycle (such as engineering drawings, charts etc.) also belong to this category. The reason for the emphasis on process modelling is only because this is a relatively new activity in enterprise design not previously practised. This modelling activity is, however, over and above the already practised ones, not to replace them

2.3.1.5.2 View Concepts

To decrease the apparent complexity of the resulting enterprise models GERA provides the view concept that allows the operational processes to be described as an integrated model, but to be presented to the user in different sub-sets (model views) of an integrated model (see Fig. 2.9). Views contain a subset of facts present in the integrated model allowing the user to concentrate on relevant questions that the respective stakeholders may wish to consider using enterprise modelling. Different views may be made available highlighting certain aspects of the model and hiding all others. The concept of view is applicable for models of all entity types across their entire life-cycle. Modelling views are generated from the underlying integrated model. Any model manipulation (any change of the contents of a particular view), will be reflected in all relevant views and aspects of the model.

GERA defines a "finest mesh of subdivision" of the kinds of models deemed desirable, allowing for the fact that an even finer subdivision may be prescribed

by a GERAM-compliant candidate architecture. The following subdivisions of models or model views have been identified in GERA:

- Entity Model Contents Views: function, information, resource, organisation;
- Entity Purpose Views: customer service and product, management and control;
- Entity Implementation View: human implemented tasks, automated tasks (mission support technology, and management and control technology);
- Entity Physical Manifestation Views: software, hardware;

Additional views may be defined according to specific user needs.
GERAM does not require every view to be present in every life-cycle phase. However, it requires that the scope of the defined views are covered by any other different view subdivision. Thereby it is guaranteed that all relevant facts are captured. For example, it is not as important to have a separate software view and separate hardware view as it is to model both software and hardware. The Enterprise Engineering Methodology decides which model to produce and which modelling language or formalism to use to describe that model. In other words the enterprise engineering process needs models for some pragmatic purpose. For example, models can be used:

a) to express a design choice;
b) to simulate a process in order to find out some process characteristics, such as cost or duration;
c) to analyse an existing process for finding inconsistencies or other problems in the information or material flow;
d) to analyse decision functions and find missing decisional roles.

The view concept is the generalisation of the view concepts of many architectures including CIMOSA, GRAI (and others). The GERA modelling framework allows for languages of different *expressive power* for each model view. This means that there is a choice of language in any particular view depending on what analysis capability (and therefore expressive power) is required, according to the enterprise engineering methodology's needs.

2.3.1.5.2.1 Entity Model Content Views Four different model content views have been defined for the user oriented process representation of the enterprise entity descriptions: Function, Information, Organisation and Resource.
The *Function View* represents the functionalities (activities) and the behaviour (flow of control) of the business processes of the enterprise. Decisional activities of the management related operations are represented, as well as transformational and support activities. The functional view of the management- and control system of an enterprise or entity is indeed the functional model of its decision system. (Note that the management- and control system of the enterprise is often called the decision system.). The function view includes functional models, process models, decisional models, which differ in

their expressive power (and competency, e.g. in terms of what analysis questions these models can answer) but all talk about some aspect of the enterprise function. As a result, the "function view" is a holding place for a host of possible models, such as CIMOSA (Vernadat , 1996, 1998) function view models, GRAI Grid (Doumeingts et al. , 1998) and GRAI Net representations of decision centres, Petri nets, Event Driven Process Chains, Generalised Process Networks, QGERT or GPSS models, ... and so on.
All of the above types of models belong to the "function" view. Similar arguments can be developed for the information, for the organisation, and for the resource view.
The *Information View* collects the knowledge about objects of the enterprise (material and information) as they are used and produced in the course of the enterprise operations. The information is identified from the relevant activities and is structured into an enterprise information model in the information view for information management and the control of the material and information flow.
The *Resource View* represents the resources (humans and technical agents as well as technological components) of the enterprise as they are used in the course of the enterprise operations. Resources will be assigned to activities according to their capabilities and will be structured into resource models e.g. for asset management.
The *Organisation View* represents the responsibilities and authorities on all entities identified in the other views (processes, information, resources). It caters for the structure of the enterprise organisation by organising the identified organisational units into larger units such as departments, divisions, sections, etc.
Other modelling views may be defined if needed (such as ecological, economic) and supported by the engineering tools.
The Entity Model Content Views in particular cover a great deal. This is because there are many different languages that fit any given model view in this category.

2.3.1.5.2.2 Entity Purpose Views Two different views allow to represent the model contents according to the purpose of the enterprise entity:
The *Customer Service and Product View* represents the contents relevant to the enterprise entity's operation and to the operation results. This represents the mission of the enterprise entity being studied.
The *Management and Control View* represents the contents relevant to management and control functions necessary to control that part of the enterprise entity that produces products or delivers services for the customer.
This view subdivision is defined to delineate the scope of the description of the enterprise, maintaining that the scope should extend to both the mission fulfilment part and the management part of the enterprise. An enterprise engineering methodology may propose that separate models or descriptions be prepared for these two parts.

2.3.1.5.2.3 Entity Implementation View The implementation of the enterprise entity may be presented in two different views based on the division between human- and automated tasks:

- The Human Activities View represents all information related to the tasks to be done by humans. The view distinguishes between the tasks that could be done by humans (extent of the tasks that are within capabilities of people) as opposed to those that will be done by humans (extent of automation).
- Automated Activities View presents all the tasks to be done by machines. This includes information related to those tasks to be carried out by mission support technology and those carried out by management and control technology (i.e. "technology tasks"). The implementation view distinguishes between the tasks which could be done by machines (extent of to tasks that are within the capabilities of machines) as opposed to those which will be done by machines (extent of automation).

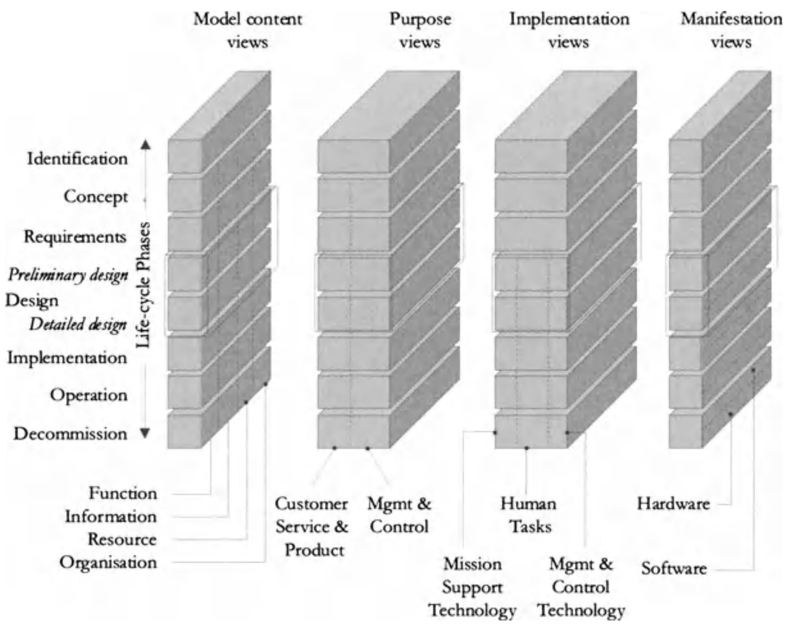


Fig. 2.9. The Modelling View concept: four essential view types and their contents. Other modelling views may be defined if needed and supported by the engineering tools.

2.3.1.5.2.4 Entity Physical Manifestation Views Again, two different views allow to represent the physical manifestation of the enterprise entity:

- The Software View represents all information resources capable of controlling the execution of the operational tasks in the enterprise. Examples are: any computer program, that can be stored in a computer or in any other control device enabling the execution of operational task, a set of instructions for humans with defined skills, such that the instructions for the humans to perform a task that they otherwise would not have been able to carry out. Software is also a controllable state, e.g. a configuration description of manufacturing hardware, such that the hardware in that configuration can perform a task provided the configuration is maintained for the duration of that task.
- The Hardware View represents all physical resources that have the capability to perform some sets of tasks in the enterprise. Examples are: a computer system with given performance characteristics, an employee with given skills, or a machinery with given functionality.

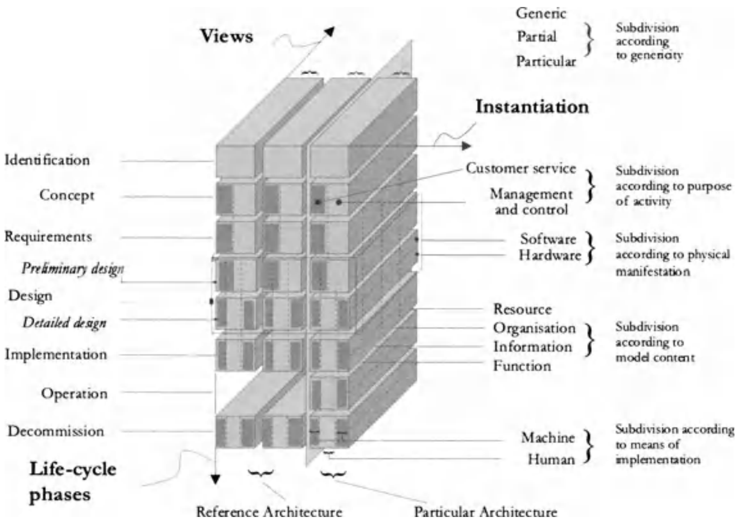


Fig. 2.10. GERA Modelling Framework with Modelling Views

Figure 2.10 shows an overlay of the different views identified above. The view categories are in general independent of each other, but certain combinations may be useful to represent specific aspects of the enterprise at particular life cycle phases. The availability of any view is subject of its implementation in the supporting engineering tool.

2.3.2 EEMs – Enterprise Engineering Methodologies

Enterprise engineering methodologies describe the processes of enterprise integration. Their scope is defined in the GERA life-cycle concept. A Generalised methodology like generalised architectures is applicable to any enterprise regardless of the industry involved.
An enterprise engineering methodology will help the user in the process of the enterprise engineering of integration projects whether the overall integration of a new or revitalised enterprise or in management of on-going change. It provides methods of progression for every type of life-cycle activity. The upper two sets of these activities (identification and concept) are partly management and partly engineering analysis and description (modelling) tasks. The requirements and design activity sets are mostly oriented towards engineering tasks throughout the process, including the production of enterprise models and designs throughout the process.
Enterprise engineering methodologies describe the process of enterprise integration and will guide the user in the engineering tasks of enterprise modelling. Different methodologies may exist that will cover different aspects of the enterprise change processes. These may be complete integration processes, or incremental changes as experienced in a continuous improvement process.
The enterprise integration process itself is usually directed to a repetitive service- or manufacturing enterprise or a project enterprise. The methodology may be specifically oriented to the type of enterprise or entity under consideration.
Enterprise engineering may itself be carried out as a specific project. But the integration task may start at any one of the enterprise's life-cycles activities, not necessarily in the top 'identification' ones. For example: a given engineering project of a new plant may not have to start with the identification and concept definition of the plant, because the customer (who commissioned the design and building of the plant) may have already carried out these activities. In this case the engineering project enterprise should only specify the requirements and carry out the design / detailed design, and implementation (building) of the plant. Such engineering project will then use the requirements, design, and implementation parts of a complete enterprise engineering methodology.
Therefore, in an enterprise engineering methodology the processes relating to the different tasks of enterprise engineering should be defined independent of each other in order to allow for their combination in the context of the particular engineering task.
Enterprise engineering methodologies may be described in terms of process models or descriptions with detailed instructions for each type of activity of the integration process. This allows not only a better understanding of the methodology, but provides for identification of information to be used and produced, resources needed and relevant responsibilities to be assigned for the enterprise engineering process, in the course of project-management of inte-

gration projects. A process representation of a methodology could employ the relevant enterprise modelling languages. Enterprise engineering methodologies may also use modelling methodologies as components. A modelling methodology is a methodology with the aim of giving help to model developers who use a modelling language or set of languages, and describes how a model can be developed and validated (starting from scratch or using pre-defined partial models).

2.3.2.1 Human Factor

The major part of a methodology is a structured approach that defines not only all the phases to be followed in an engineering and/or integration project, but also the way of involving as much as possible people working in the company (users) in the analysis and design of the manufacturing and service system.

The involving of company users is an important success factor for an integration project. It is considered that techniques used to build new manufacturing and service systems are currently difficult to understand for business users of the future system particularly in the domain of the Information Technology. Besides, according to the amount of investment necessary to build a new manufacturing and service system, one needs to be sure that the design solution of the new system meets the objectives defined in the initial user requirements. The design of the new system must be validated by users before any development or implementation.

The involving of people of the company will facilitate the final acceptance of the designed system and thus shorten the transition phase between the old and new systems. The methodology should make clear distinction between the two major phases of the design : User oriented design and technology oriented design. The experiences show that business people must be associated as much as possible in the user oriented design phase and as little as possible in detailed technology oriented design because it is an expert task (unless the technology considerations have a direct business effect).

The other aspect of human involvement in the enterprise is the place of humans in the designed entity, such as a plant.

To show the true place of the human in the implementation of the enterprise functions, there is a need to assign the appropriate ones of these tasks and functions developed in the Requirements Life-cycle Phase to the human element of the system. This can be done by considering the functional tasks as grouped in three boxes in the Preliminary Design Phase. (see Fig. 2.11)

This action will separate the tasks of Mission Fulfilment and Management and Control as defined in the Requirements Analysis phase into three. Thus, the tasks or functions involved are assigned to the appropriate boxes that in turn define the automated information tasks that become the Management and Control Information Systems Architecture functions, and the automated manufacturing and service tasks that become the Mission Support Equipment

Architecture functions. The remainder (non-automated) become the functions carried out by humans as the Human and Organisational Architecture.
The split of functions for implementation between humans and machines forms the first definition of the implementation of the resulting manufacturing system. Because of the inclusion of humans, there must be three separate elements in the implementation scheme: the Management and Control Information System Architecture, the Human and Organisational Architecture, and the Mission Support Equipment Architecture.
Two lines, describe the extent to which automation is possible ('Extent of Automatability Line') and the extent to which human implementation is possible ('Extent of Humanizability Line'), can be defined giving the limits of automation and the limits of human involvement. The 'Automatability Line' shows the absolute extent of pure technologies in their capability to actually automate the tasks and functions. It is limited by the fact that many tasks and functions require human innovation, and so forth, and cannot be automated with presently available technology.
The 'Humanizability Line' shows the maximum extent to which humans can be used to actually implement the tasks and functions. It is limited by human abilities in speed of response, breadth of comprehension, range of vision, physical strength, and so forth.

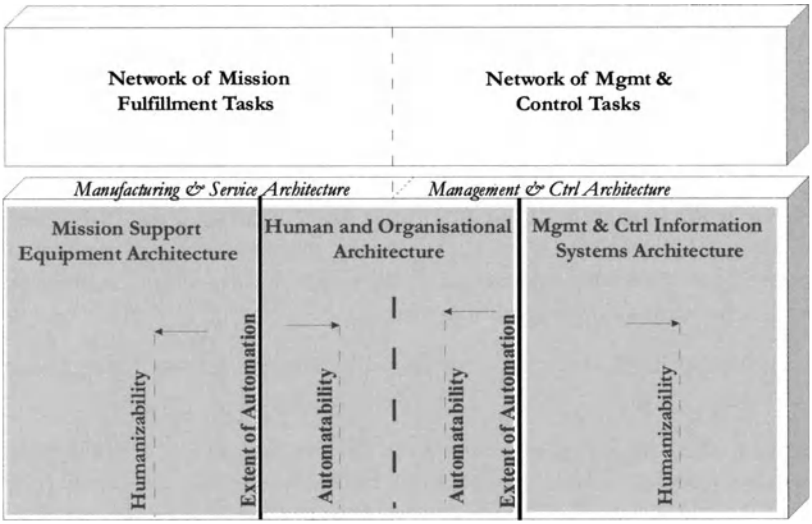


Fig. 2.11. Introduction of the Concepts of Automatability, Humanizability and Extent of Automation Lines to Define the Three Implementation Architectures

Still a third line, the Extent-of-Automation Line, can be drawn that shows the

actual degree of automation carried out or planned in the subject enterprise entity. Therefore, it is the one which actually defines the boundary between the Human and Organisational Architecture and the Management and Control Information System Architecture on the one side, and the boundary between the Human and Organisation Architecture and the Mission Support Equipment Architecture on the other side. Provided requirements such as timing and co-ordination are fulfilled, it makes no difference what functions are carried out by personnel versus machines, or what organisational structure or human-relations requirements are used. Therefore, the actual extent of automation is determined by political and human relations-based considerations as well as by technical ones. The location of the Extent of Automation Line is influenced by economic, political, social (customs, laws and directives, union rules), as well as technological factors.

2.3.2.2 Project Management

In order to perform in an efficient way the analysis, the design and the implementation within an engineering and/or integration project, the methodology must associate with the available project management techniques in terms of project planning, project budgeting and control, project follow-up and so forth.
A logical separation can be made between a Project life-cycle and Enterprise System life-cycle (see 2.3.1.3.3). Within the project life-cycle:

a) the control of the project is covered by the "management and control" part of the project life-cycle, and
b) the execution (operation) of the project is covered by the "service to the customer" part, as guided by the various phases defined in the life-cycle of the system that is designed / built by the project. In this sense one of the main activities within the project management's operation is the planning of time and resources and the control of the steps to be executed and defined in the system life-cycle.

Looking at the life history of a project it contains at least three phases in time:

- project start-up, aimed at defining the project organisation (various teams and managers), project preparation (definition of the what, who, when and how), project planning and the organisation of the project start-up meeting;
- project control, aimed at acceptance of deliverables (hard and/or software, machines, various installations...), monitoring of progress and continuous planning, managing problems and change, and executing reviews and auditing;
- project termination, aimed at the general acceptance and the final evaluation of the project.

Examples of project management approaches associated with a methodology can be found in GIM, SADT etc.

2.3.2.3 Economic Aspects

A methodology must take the economic aspect into consideration. In fact, the choice of various investments depends on objectives that are often contradictory. To help designer to choose the best solution, both technical and economic views should be studied at the different steps of an integration project.
The methodology should allow the decomposition of the strategic objectives of the company into sub-objectives of each function; and the specification of the technical solution must be followed by a technical-economic evaluation. The economic evaluation can be split up in 3 steps:

- calculation of the cost of the solution,
- performance measures of the solution,
- comparison of the solution costs with the budget.

The aim of this approach is on the one hand, to compare the project cost against the investment budget, and on the other hand, to compare the solution performances against the technical objectives derived from the company strategy. This comparison will allow to economically validate or not the proposed solution.
Examples of technical-economic evaluation approach can be found in [PB1] GEM (GRAI Evolution Methodology), ECOGRAI or Activity Based Costing.

2.3.3 EMLs – Enterprise Modelling Languages

The engineering of an enterprise is a highly sophisticated, multidisciplinary management, design and implementation exercise during which various forms of descriptions and models of the target enterprise will be created.
To develop enterprise models potentially more than one modelling language is needed. The situation is similar to software engineering where there are no known languages that span the needs of all models in all phases of the life-cycle. The set of languages must be competent to express the models of all areas defined in the modelling framework of the Generalised Enterprise Reference Architecture, GERA.
Enterprise models must represent the enterprise operations from various modelling viewpoints (see 2.3.1.5.2). For each area of the GERA modelling framework, there may be a modelling language selected according to the enterprise engineering methodology, which is suitable for the expression of the respective models. In practice, the set of languages will be smaller then the set of areas to be modelled, with one language suitable for more then one area.
Two requirements must be satisfied in the definition of a complete set of enterprise modelling languages:

- every area represented in the modelling framework (see Fig. 2.9 and Fig. 2.10) must be covered for every enterprise entity type;
- a model developed in one subject area must be able to be integrated with models of other subject areas, if the information content of the model so requires.

Any subject area of modelling may be covered with more than one language, the languages being of different 'expressive power', meaning that some languages may only be useful for the description of the subject area but not suitable for certain analysis tasks. For example, the languages that belong to the function view may differ in their capability of expressing certain characteristics of functions: for example the dynamics of the function, the behaviour of the function, or the subdivision of the function into function types such as product management, resource management, and co-ordination and planning. The necessary expressive power, and thus the selection of languages, is related to the methodology followed.
An enterprise engineering methodology may prescribe some analysis tasks that require a given modelling language. However, there should not be any prejudice built into the modelling languages as to what the methodology will be. It is necessary for any enterprise engineering methodology to have access to a consistent set of modelling languages for typical enterprise engineering tasks. Therefore, such consistent and complete sets must be selected or developed, e.g. CIMOSA set of languages, the choice of the set of languages by the GRAI methodology, etc.
Enterprise modelling languages may be defined as modelling constructs. Modelling constructs represent the different elements of the modelled enterprise entity and improve both modelling efficiency and model understanding. The form (representation) of modelling constructs has to be adapted to the needs of people creating and using enterprise models. Therefore, separate languages may exist to accommodate the aspects of different users (e.g. business users, system designers, IT-modelling specialists, and others). In addition, modelling languages may allow the formation of higher level constructs out of more basic constructs (macro constructs) to enhance modelling productivity.
Model based decision support and model driven operation control and monitoring require modelling constructs that are supporting the end users. They have to represent the operational processes according to the users' perceptions. The semantics of the modelling languages may be described in terms of ontological theories (see 2.3.4.3). This is especially important if enterprise models are to support the enterprise operation itself, because the models in that case must be executable. However, the definition of the formal semantics should be supported by natural language explanations of the concepts as well.
Examples of modelling languages can be found in ARIS (Scheer , 1998), CIMOSA (Vernadat , 1996, 1998), GRAI/GIM (Doumeingts et al. , 1998), IEM (Spur et al. , 1993) or the IDEF family of languages (Menzel and Mayer , 1998; Vernadat , 1996). Relevant Standardisation: ENV 12 204 defines a ref-

erence set of twelve Constructs for Enterprise Modelling (Business Process, Capability Set, Enterprise Activity, Enterprise Object, Event, Object View, Object State, Order, Organisational Unit, Product, Resource, Relation,). ISO DIS 14258 defines Rules and Guidelines for Enterprise Models.

2.3.4 GEMCs - Generic Enterprise Modelling Concepts

Generic Enterprise Modelling Concepts are the most generically used concepts and definitions of enterprise integration and modelling. Three forms of concept definition are, in increasing order of formality:

- Glossaries
- Meta-models
- Ontological theories.

Some requirements that must be met are as follows:

- Concepts defined in more then one form of the above must be defined in a mutually consistent way;
- Those concepts that are used in an enterprise modelling language must also have at least a definition in the meta-model form, but preferably the definition should appear in an ontological theory

2.3.4.1 Glossary

The terminology used in enterprise integration can be defined in natural language as a Glossary of Terms. Such a Glossary is a mandatory requirement for a complete generalised enterprise integration architecture and methodology. As a minimal requirement the glossary must define all concepts that are defined in the semi-formal meta-models and/or formal ontologies.

2.3.4.2 Meta-models

In the GERAM context, meta-models are conceptual models of the terminology component of modelling languages[15]. They describe the concepts used, their properties and relationships, as well as some basic constraints, such as cardinality constraints.
Meta-models are situated between informal and formal expressions. Normally, they are represented as an entity relationship schema or in a language similar in expressive power. The terminology defined in the integrated meta-schema may also be considered as the schema (at any one time) of an enterprise engineering tool's database of enterprise models.

[15] Meta-models are models about models.

2.3.4.3 Ontological Theories

Ontological theories are formal models of the concepts that are used in enterprise representations. They capture rules and constraints of the domain of interest, allowing useful inferences to be drawn, to analyse, execute (e.g. for the purposes of simulation), cross check and validate models.
Ontological theories are a kind of generic enterprise model, describing the most generic aspects of enterprise-related concepts (function, structure, dynamics, cost, and so forth.), and define the semantics of the modelling languages used. They play a similar role to what 'data models' play in database design.
Enterprise modelling languages backed by ontological theories (and their supporting enterprise engineering tools) provide the user with enhanced analysis capabilities.
Since separate enterprise modelling languages may be used to describe various aspects / views of the enterprise it is important to stress that the ontological models must be integrated, i.e. the language definitions for views should be views of an integrated meta-schema (if such a meta schema is defined) and/or of its underlying ontological model (if the corresponding ontological theory is defined). This purely technical requirement allows enterprise engineering tools to be used to cross-check the mutual consistency of enterprise models produced in the enterprise engineering process.[16]

2.3.5 PEMs – Partial Enterprise Models

Partial enterprise models (reusable reference models) are models that capture concepts common to many enterprises. PEMs will be used in enterprise modelling to increase modelling process efficiency. In the enterprise engineering process these partial models can be used as tested components for building particular enterprise models (EMs). However, in general such models still need to be adapted (completed) to the particular enterprise entity.
Partial models may be expressed as:

- Models that capture some common part of a class of enterprises,
- Paradigmatic (reference or prototypical) models that describe a typical enterprise of a class. Prototype models can be subsequently modified to fit a particular case,
- Abstract models of a part or whole of a class of enterprises that capture the commonalities but leave out specific details. This type of model is of the 'fill-in-the-blank' type.

2.3.5.1 Partial Human Role Models

Needed are partial models on human roles in decision making, on professional capabilities and skills, on socio-technical aspects (motivation, incentives, and

[16] There are theoretical limitations to this cross-checking, so the wording really should be: to cross check to the best possible extent.

so forth). Related partial models will be on enterprise organisation and the identification of human responsibilities and authorisation in those models.

2.3.5.2 Partial Process Models

The provision of reusable reference models of business processes can significantly improve the efficiency of enterprise modelling. These models represent a common view of the enterprise's operational processes and are concerned with various processes, such as purchasing processes, order processes, product development processes, administrative processes (representing workflow procedures or CSCW), relations with external organisations (e.g. banks).

Partial process models could be tailored to specific industries or industry types (like automotive, electronic components industry, or more specific industries, such as car suspension manufacturing, VLSI manufacturing, and so forth) or the models may represent typical management and control systems, such as various forms of enterprise co-operation. For example, the modern paradigms of extended and virtual enterprises could be represented as partial models guiding interested parties in defining their specific forms of co-operation.

It is to be noted, that these models of business processes would typically use one or more forms of model view (see 2.3.1.5.2), such as function and behaviour models, database schemata, and so forth. Typically partial process models would describe common functionality but leave the definition of the process behaviour to the decision of the particular enterprise.

Partial models may be presented on various levels of abstraction and using various model content views. For example, ISO 9000 quality models are partial models, defining typical or required policies that must be adopted by quality-accredited companies (some ISO 9000 standards go further and define in more detail certain aspects of the business process functions). Many companies create partial models in form of company-wide database schemata that are then enforced in all company databases, or are used as a basis for such designs. (Such common database schemata can be used as standard interfaces between processes). Some partial models are provided as a system of model-fragments that ensure that the resulting models define a high-quality, business-process model as well as feasible system implementation.

2.3.5.3 Partial Technology Models

Partial technology models will provide common descriptions of resources and their aggregations like shop floors, assembly lines, flexible manufacturing systems, office systems, IT systems, etc. All of these partial models will most probably be industry and/or country specific, providing common descriptions of components (linked to supplier catalogues), common operational rules, etc.

2.3.5.3.1 Partial Models of IT systems

Partial models of IT systems can be all the components needed in communications and information processing, that will guide and enhance the design of information systems. This includes the enabling technology for enterprise integration (EDI, STEP, HTML/WWW, etc.)
One important partial model commonly needed for enterprise integration is the one of integrating services (see 2.3.1.4.1) that provide portability across heterogeneous environments. These services have to include communication, processing/execution control, presentation and information services. The definition of such services should itself refer to enabling standard definitions, such as EDI; STEP; HTML/HTTP and all other communication protocols; CORBA-IDL; SQL3; Java services for execution, compilation, presentation etc.).
Such services can then be packaged in various ways as modular products or building blocks (see 2.3.7).

2.3.6 EETs – Enterprise Engineering Tools

Enterprise engineering tools will deploy enterprise modelling languages in support of enterprise engineering methodologies, and specifically must support the creation, use, and management of enterprise models.
Enterprise engineering tools must support the analysis and evaluation of the models (or descriptions) of the enterprise, and its products, as well as allow the enactment of the models for simulation. These functions are needed for design decision making in the course of enterprise engineering.
Engineering tools should provide user guidance for the modelling process and provide useful analysis capabilities for the use of the models in the enterprise engineering process. That is, the tools help the user utilise the models for the advancement of the engineering process to the best possible extent, as well as releasing the models for operation to support decision making and model based operation monitoring and control.
An enterprise engineering tool is required to support the simultaneous design / engineering activity of the enterprise entity in question. Therefore it needs to:

- support collaborative as well as individual design / engineering activity
- provide a shared design repository, or database, that allows the management of all partial and particular models and descriptions that are used or produced in the enterprise engineering process, including formal models and any other informal design descriptions, document, etc.

Depending on the enterprise entity in question these engineering tools of the enterprise may display a great variety. If the object of design is a project (i.e. project enterprise) or an enterprise (such as a company) then the tools will be supporting the creation of the design of such enterprise, including its business

processes, resources, organisation, etc. If the enterprise entity in question is a product, or product type, then the tools will be supporting the design of the product, such as functionality, geometry, control system, operator procedures, and so forth.

Through the potential integration of the enterprise engineering and model execution services there is scope for the engineering services to be interconnected with enterprise management services. (For example, the initial simulation of a project's execution may use similar tools to those that are utilised for continuous planning of the project during project execution.)

Engineering tools should enable the user to connect the models with the real business process, so as to keep the models up-to-date. Note that engineering tools may be either separate or integrated with the model execution environment (see 2.3.7).

The ideal engineering environment should be modular so that alternative methodologies could be based on it. Therefore, engineering tools should provide an environment that is extensible rather than be based on a closed set of models, leaving space for alternative modelling methods (e.g. through enriching the modelling language constructs, or adding new views, as appropriate).

Some examples of engineering tools based on modelling languages (when the enterprise entity in question is the enterprise or a project enterprise itself) are as follows: ARIS Toolset (ARIS) (Scheer , 1998), FirstSTEP (CIMOSA), MO^2GO (IEM) (Mertins and Jochem , 1998), KBSI Tools (Tissot and Crump, 1998), METIS, Processwise, etc. There are many examples for engineering tools of the enterprise for the case when the enterprise entity in question is a product; such tools include tools for product modelling and design, simulation, visualisation, control systems design, and so forth; e.g. STEP Tools.

2.3.7 EMOs - Enterprise Modules

Enterprise modules are implemented building blocks or systems (products, or families of products), that can be utilised as common resources in enterprise engineering and enterprise integration. As physical entities (systems, subsystems, software, hardware, available human resources/professions) such modules are accessible in the enterprise, or can be made easily available from the market place. In general EMOs are implementations of partial models identified in the field as the basis of commonly required products for which there is a market. Enterprise modules may be offered as a set, such that if the design of the enterprise is following the partial models that form the basis of this set, then the resulting particular enterprise's business system can be implemented using some or all modules of this set of modules. One set of enterprise modules of distinguished importance is the Integrating Infrastructure that implements the required Integrating IT Services (see 2.3.5.3.1).

2.3.8 EMs – Enterprise Models

The goal of enterprise modelling is to create and continuously maintain a model of a particular enterprise entity. A model should represent the reality of the enterprise operation according to the requirements of the user and his application. This means the granularity of the model has to be adapted to the particular needs, but still allows interoperability with models of other enterprises. Enterprise models include all those descriptions, designs, and formal models of the enterprise that are prepared in the course of the enterprise's life history.
Enterprise models are expressed in enterprise-modelling languages and are maintained (created, analysed, stored, distributed) using enterprise engineering tools. Both model creation and model use should be supported by real-time information services. The use of such services will ensure access to real time information in both enterprise environments, the engineering and the operational one.
Some important uses of enterprise models are:

- decision support for evaluating operational alternatives in the enterprise engineering process (enabling operation analysis and capturing the results of synthesis);
- communication tool that enables the mutual understanding of issues between stakeholders of the enterprise, both internal and external ones;
- model driven operation control and monitoring, for efficient business process execution,
- training of new personnel, where enterprise models serve as demonstration of the real business process for new employees.

2.3.9 EOSs – Enterprise Operational Systems

Enterprise Operational Systems support the operation of a particular enterprise. They consist of all the hardware and software needed to fulfil the enterprise objective and goals. Their contents are derived from enterprise requirements and their implementation is guided by the design models that provide the system specifications and identify the enterprise modules used in the system implementation.

2.4 Historical Note

In 1992 IFIF and IFAC established a joint Task Force to review existing approaches to Enterprise Integration and to make recommendations to the industrial and research community.
The IFIP constituent is the Task Force is operating under WG 5.12 of TC5, the IFAC constituent is operating under the Coordinating Committee 'Manufacturing and Instrumentation, TC-MIA.

The Task Force was chaired by Prof Emeritus T.J.Williams (Purdue University) from 1992 till 1996, and by A/Prof Peter Bernus (Griffith University) from 1996 till 2002.
Members comprised representatives from the industrial and research communities, with several researchers coming from previously held management or consultant positions in industry.
Enterprise Integration has steadily evolved from the ninteen-seventies with increasing needs of integrating the information and material flow throughout the enterprise. Separate achievements have been accomplished in the area of manufacturing both in design and production (NC and CAD/CAM systems, CIM systems, Manufacturing Cells, Material Requirements Planning and Production Scheduling Systems), and in the area of business support (Accounting, Financial Planning, Human Resource Management, Decision Support Systems, etc).
By the mid-eighties it has become clear that isolated efforts lead to systems that can not easily communicate and thus elaborate islands of automation had to be maintained, where the integration of these proved to be difficult. Today, industry still feels the problems arising from this, with many isolated 'legacy' applications being still in use (although the renewal programs associated with the Year 2000 problem have considerably eased the pain).
At the same time, in the mid-eighties, it transpired that considering exclusively the automated parts of material and information processing is no longer tenable, because the human element in the enterprise is still the most important part, and thus an approach is needed that deals at the same time with the human and automated parts of the enterprise. Thus the complete enterprise, as any other human made system, needs to be properly designed, and there is a need for methods to do so.
Two approaches have emerged to respond to this challenge.
The first approach was based on generic models, or designs, (called 'architectures') that could subsequently be implemented as information systems products (or families of products), incorporating most or all information processing tasks in the enterprise (especially its management). The resulting systems were called Enterprise Resource Planning (ERP) systems. Also, specifically for CIM systems, a number of CIM Reference Models were developed, that tried to systematise the functional building blocks of a CIM system. The problem was, however, that the number of competing models was in the order of several dozens, with all of these failing to achieve an industry-wide acceptance, or standard status. The appeal of this approach was that it could easily be turned into products (software systems).
The second approach was based on the recognition that similarly to many engineering disciplines (such as chemical engineering, manufacturing engineering, software engineering, civil engineering, etc) 'enterprise engineering' should also be based on the so-called life-cycle approach. According to this approach, in order to create an integrated enterprise the enterprise creation activities (and thus methodologies) must extend to the whole of life of the enterprise from its

inception till it is no longer operated (i.e. when it is decommissioned). Several such architectures were developed - some by groups with manufacturing systems background, and some with information systems background.

References

Bernus, P. and Nemes, L., A Framework to Define a Generic Enterprise Reference Architecture and Methodology, Proceedings of the International Conference on Automation, Robotics and Computer Vision (ICARCV'94), Singapore, November 10–12, (1994). Also in *Computer Integrated Manufacturing Systems* 9,3 (July 1996) pp 179-191.

Bernus, P., Mertins, K. and Schmidt, G.,(Eds). Handbook on Architectures of Information Systems, Springer-Verlag., (1998).

CIMOSA - Open System Architecture for CIM; ESPRIT Consortium AMICE, Springer-Verlag, Berlin, 1993, (ISBN 3-540-56256-7), (ISBN 0-387-56256-7).

CIMOSA - Open System Architecture for CIM, Technical Baseline; Version 3.2 CIMOSA Association, private publication (1996).

Doumeingts, G., Vallespir B. and Chen, D., Decisional Modelling using the GRAI Grid, in Bernus, P.,Mertins, K. and Schmidt, G. (Eds), Handbook on Architectures of Information Systems, Springer-Verlag., (1998). pp313-338

Fox, M.S. "The TOVE project: towards a common-sense model of the enterprise", Proc ICIEMT92, Petrie, C.J., Jr. (Ed), MIT Press, (1992) pp 310-19

Kosanke K., "Comparison of Enterprise Modelling Methodologies", Proceedings DIISM'96, Chapman & Hall, London, (1997).

Kosanke, K., Nell, J., (Eds), Proceedings of the International Conference on Enterprise Integration and Modelling Technology, Springer-Verlag, (1997).

Menzel, C. and Mayer, R.J., The IDEF Family of Languages, in Bernus, P.,Mertins, K. and Schmidt, G. (Eds), Handbook on Architectures of Information Systems, Springer-Verlag., (1998). pp209-242

Mertins, K. and Jochem, R., MO^2GO, in Bernus, P.,Mertins, K. and Schmidt, G. (Eds), Handbook on Architectures of Information Systems, Springer-Verlag., (1998). pp589-600

Petrie, C. J., Enterprise Integration Modelling; ICEIMT Conference Proceedings, The MIT Press, Cambridge, MA,(1992) (ISBN 0-262-66080-6).

Scheer, A.W., ARIS, in Bernus, P.,Mertins, K. and Schmidt, G. (Eds). Handbook on Architectures of Information Systems, Springer-Verlag., (1998). pp541-566

Shorter, D. (Ed), An evaluation of CIM modelling constructs - Evaluation report of constructs for views according to ENV 40 003, *Computers in Industry* - Vol. 24, Nrs 2-3 (1994).

Spur, G., Mertins, K. and Jochem, R., Integrierte Unternehmensmodellierung. Beuth Verlag. Berlin. 1993.

Tissot, F. and Crump, W., A Truly Integrated Enterprise Modeling Environment, in Bernus, P.,Mertins, K. and Schmidt, G. (Eds). Handbook on Architectures of Information Systems, Springer-Verlag., (1998).

Vernadat, F. B., "Enterprise Modelling and Integration: principles and applications", Chapman & Hall, London, (1996), (ISBN 0 412 60550 3).

Vernadat, F.B., The CIMOSA Languages, in Bernus, P.,Mertins, K. and Schmidt, G. (Eds), Handbook on Architectures of Information Systems, Springer-Verlag., (1998). pp243-264

Weston, R.H. "Workbenches and reference models for enterprise engineering",. Chapter in Handbook of Life Cycle Engineering: Concepts, Tools and Techniques, Edited by A. Molina, J.M. Sanchez, A. Kusiaka, Chapman & Hall, London, (1998).

Williams, T.J. et al., Architectures for integrating manufacturing activities and enterprises, *Computers in Industry* - Vol. 24, Nrs 2-3 (1994).

Williams, T.J., The Purdue Enterprise Reference Architecture, *Computers in Industry*, Vol. 24, No. 2-3, pp. 141-158., 1994.

Williams, T.J. and Li, H., A Specification and Statement of Requirements for GERAM (The Generalised Enterprise Reference Architecture and Methodology) with all Requirements illustrated by Examples from the Purdue Enterprise Reference Architecture and Methodology PERA, Research Report 159, Purdue Laboratory for Applied Industrial Control, (November 1995), Version 1.1.

Standards

CEN/TC310 "CIM Systems Architecture - Enterprise model execution and integration services - Evaluation report" CEN Report CR: 1831 :1995.

CEN/TC310 "CIM Systems Architecture - Enterprise model execution and integration services - Statement of Requirements" CEN Report CR: 1832:1995.

ENV 12204 Advanced Manufacturing Technology - Systems Architecture - Constructs for Enterprise Modelling CEN TC 310/WG1, 1995.

ISO 14258:1998 Industrial automation systems – Concepts and rules for enterprise models (corrections: ISO 14258:1998/Cor 1:2000)

ISO/TC184/SC5/WG1ISO 15704:2000 Industrial automation systems – Requirements for entreprise-reference architectures and methodologies

ISO/TC184/SC5/WG1 - CEN TC 310/WG1. EN/IS 19439 : 2002. Framework for Enterprise Modelling.

[PB1]ECOGRAI overhung the typing area; I changed the order in the list – hopefully this fixes the overhanging word problem

3

A MAPPING OF INDIVIDUAL ARCHITECTURE FRAMEWORKS (GRAI, PERA, C4ISR, CIMOSA, ZACHMAN, ARIS) ONTO GERAM

Ovidiu Noran

Griffith University (Australia)

3.1 Introduction

This Chapter presents the mapping of selected type 2 (life cycle) enterprise architectures and architecture frameworks onto the Generalised Enterprise Reference Architecture and Methodology (GERAM) framework. It covers aspects regarding the reference architectures (including life cycle / life history concepts and modelling frameworks) and associated modelling methodologies, languages and reference models where applicable. The Chapter builds on previous mapping efforts of established frameworks, giving a refined interpretation of those results. In addition, several emerging frameworks are also reviewed and mapped against the GERAM.

This is done in the hope that this Chapter (and the whole Handbook) will help enterprise architects to focus on the deliverables that a change process requires, rather then conduct emotionally (and commercially) charged discussions about the pros and cons of particular frameworks. At the same time, the investment on behalf of companies and government agencies needs to be protected. Therefore, the user of a chosen particular architecture framework can use GERAM to check if the framework in question is equipped with all necessary components. Should the chosen framework prove to lack certain concepts (or contain insufficient detail to provide enough guidance towards ensuring common understanding (on behalf of all stakeholders) of deliverables in the enterprise's change processes), the user may again use GERAM to complete it[1].

The main aim of the Generalised Enterprise Reference Architecture and Methodology (as described in Chapter 2 and (ISO/TC184/SC5/WG1 , 2000a)) is to generalise the contributions of various existing and emerging Enter-

[1] i.e. the chosen framework may be added concepts, elements, views, etc of other reference architectures or frameworks.

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

prise Architecture Frameworks and Enterprise Reference Architectures[2] in order to define a complete collection of tools, methods and models to be employed by any enterprise engineering and integration effort. As such, GERAM assists in the choice of tools and methodologies by providing criteria that need to be satisfied, rather than trying to enforce particular options. Used as a generalisation of frameworks, GERAM may also assist in establishing the completeness and suitability[3] of frameworks proposed to form the basis to a particular change process[4]. Concisely, GERAM is "a tool-kit of concepts for designing and maintaining enterprises for their entire life history" (ISO/TC184/SC5/WG1 , 2000a) [5].

There have been several notable attempts to map the existing life cycle architectures[6] and their associated artefacts against one another (such as in (Williams et al. , 1996)), which have highlighted some of the difficulties encountered in the mapping process.

Subsequently, attempts were made to map the architectures against a fixed (and preferably neutral) reference, able to accommodate all possible *types of* artefacts potentially contained in the mapped architectures. This reference has been constructed by essentially combining all the features of the main existing architectures, filtering out coverage overlaps and adding any missing (but perceived as necessary) aspects[7]. The result of the mappings was a matrix-like structure of requirements (Bernus et al. , 1996c). This structure has further improved the mapping process but the result was quite complex and therefore not easy to follow.

To simplify the flat matrix-like modelling framework, a more space-efficient and user-friendly, three-dimensional structure was proposed by Williams and Li[8]. The modelling framework was then supplemented with specific concepts such as entity type, recursion, life history, etc. The architecture framework was subsequently obtained by putting together the generalised modelling frame-

---

[2] the term 'reference architecture' hereafter used interchangeably with 'modelling framework' for simplicity, although reference archtectures (such as GERA, as described in Chapter 2) may also include other concepts, such as 'life history'.

[3] i.e. used as a checklist to identify potential gaps / uncovered areas.

[4] since management may chose to combine the elements of more than one framework and use these in combination.

[5] for further details on GERAM refer Chapter 2.

[6] in the following, the terms 'life cycle architecture' and 'architecture framework' will be used interchangeably and with the meaning of the complete set of artefacts provided by an architecture framework, as shown in Fig. 3.1.

[7] inevitably, some areas are typically well covered and understood in all frameworks, while others are not (for reasons relating to (computer integrated) manufacturing and information systems history, particular frameworks purposes and intended audiences, the underlying frameworks' ontologies and so on). Typical examples are function and information (well understood in all frameworks) vs. human and decisional aspects (not so well understood in all frameworks).

[8] the proposal was presented at meetings of the IFIP-IFAC Task Force.

work and all essential generic concepts of enterprise engineering / modelling - such as enterprise models, modelling languages, generic enterprise modelling concepts, partial models, etc.
The current outcome of these efforts is a reference architecture framework reflected in the GERAM document as described in (ISO/TC184/SC5/WG1 , 2000a)[9] and Chapter 2.

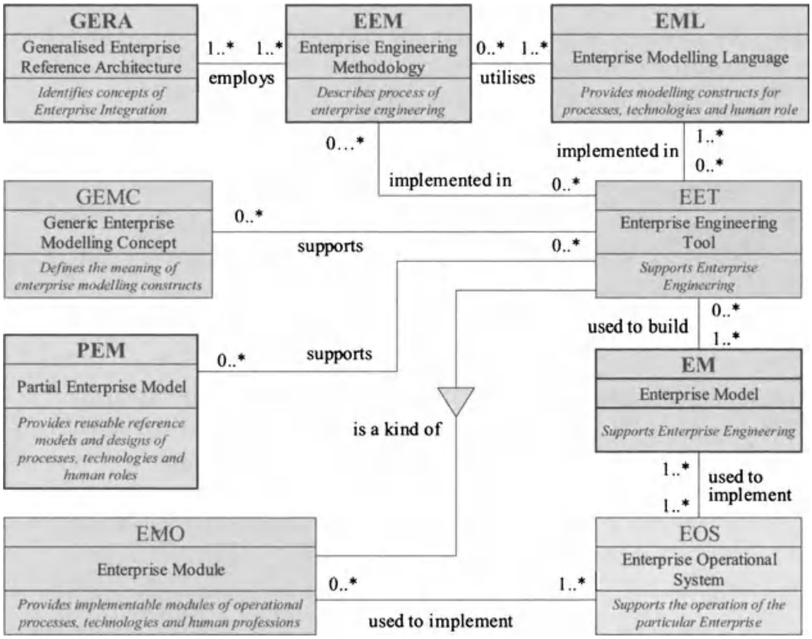


Fig. 3.1. A possible GERAM metamodel [10](based on Annex A of ISO/IS 15704 (ISO/TC184/SC5/WG1 , 2000a)).

GERAM, as a generalised architecture framework, aims to categorize all[11] life

[9] this is Annex A of ISO/IS 15704 (ISO/TC184/SC5/WG1 , 2000b), listing the requirements for enterprise reference architectures and methodologies.

[10] as such, a model about the GERAM models. **NB** Historically the abbreviation GERAM means a reference architecture and methodology, however, as it turns out in the use of GERAM (and the figure), only one component of the Framework, namely GERA is the life-cycle architecture, another one is the Methodology, and there are other important components as well (as shown in the figure)

[11] new life cycle architectures and architecture frameworks emerge and existing architectures and frameworks evolve. As a generalised reference architecture, GERAM itself is also subject to refinement and modifications in order to keep up with the evolution of the enterprise integration field (hence also Chapter 2, containing the current version of GERAM).

cycle architectures with their associated artefact types (methodologies, reference models, ontologies, etc)[12]. A consequence of the GERAM architecture and purpose is that one cannot employ GERAM alone to engineer an enterprise; GERAM simply lacks the necessary content. GERAM should however be used to assess candidate architectures for a given enterprise integration task (or task type) and enable users to make an informed decision on which combination of architectures and their components should be used so that all *necessary* aspects are covered[13].
The analysis in this Chapter will be restricted to the reference architectures and architecture frameworks identified in the title of this Chapter and will follow the GERAM metamodel shown in Fig. 3.1.
Aspects covered for each life cycle architecture reviewed include:

- the reference architecture (including life cycle phases and life history aspects);
- modelling languages proposed in the modelling framework (if applicable)[14];
- supported modelling methodologies;
- reference models.

For the sake of completeness, other important aspects identified in the GERAM framework (such as generic modelling concepts, enterprise modules, modelling tools, etc) relating to the reviewed architectures are also briefly covered[15]. Enterprise Operational Systems (EOSs) support the operation of a particular enterprise and as such are not of particular interest for the topic covered by this Chapter.

3.2 Life Cycle Phases

Mainstream system engineering and enterprise integration literature acknowledges the existence of two main types of architectures: architectures that represent the structure of a system at a given point in time (snapshot) and *life*

[12] as such, GERAM may resemble an empty bookcase providing shelves for e.g. all possible books in the field of enterprise integration, or a to-do / shopping list for a given enterprise integration task.

[13] GERAM specifies that not all views contained within its GERA modelling framework should necessarily be explicitly defined in all life cycle phases, as long as the *scope* of these views *is* covered one way or another.

[14] not all modelling frameworks propose a particular set of modelling languages. Those that do not (e.g. PERA, partially Zachman), delegate the selection of modelling languages to the user of the framework. Those that do (CIMOSA, GRAI-GIM), do so with a type of a enterprise integration task in mind (typically widespread tasks, making these frameworks very generic in their applicability).

[15] for more details on these aspects, not necessarily related to the reviewed architectures, please refer to other Chapters in this book.

cycle architectures, which describe the possible phases and artefacts involved in the existence of a system[16].
An essential part of the requirements for a life cycle architecture is to include the concept of *life cycle phase*, understood as a set of possible processes or activities[17] which *may* be performed (once or several times) in the enteprise during its existence (i.e. throughout its *life history*).
Common misconceptions are based on a lack of consistent terminology across the field[18] and may easily lead to the erroneous conclusion that life cycle phases imply succession, i.e. a temporal aspect. The life cycle dimension in the GERA sense[19] abstracts from time and it is simply a repository of possible processes performed in the life history of a system, categorised according to the level of abstraction that these processes use to consider the enterprise entity in question[20]. Some of these processes may be performed repeatedly and in different succession and some may not occur at all during the existence of the system. For a concept that *does* take into account time and succession, refer to Section 3.3 in this Chapter and to Section 2.3.1.3.2 within Chapter 2.

3.2.1 PERA and the Life Cycle Concept

Historically, the Purdue Enterprise Reference Architecture (PERA, (Williams , 1994a)) represents the model of an enterprise engineering methodology.
Since the PERA diagram employs the same temporal abstraction of methodological steps as GERAM[21], a 'direct' life cycle phases mapping is possible (refer Fig. 3.2). The Purdue Guide for Master Planning (the PERA methodology) separately elaborates on typical iterative and successive application of these phases, as potentially encountered in an enterprise engineering project. From the point of view of GERAM, PERA is one of the most complete life cycle architecture in its coverage of the life cycle phases.

---

[16] such as conception, development, build, operation, dissolution etc.
[17] a process may be a "partially ordered set of enterprise activities" (ISO/TC184/SC5/WG1 , 2000b).
[18] several such problems were described and solutions proposed in (Noran , 2000a)- e.g. phase vs. stage, life cycle vs. life history, etc
[19] refer Chapter 2 for details.
[20] e.g. the design process may consider the enterprise entity in question on a different level of abstraction than e.g. the identification process
[21] PERA is one of the main life cycle architectures leading to the creation of GERA, contributing (among others) the Identification, Concept, Operation and Decommission life cycle phases.

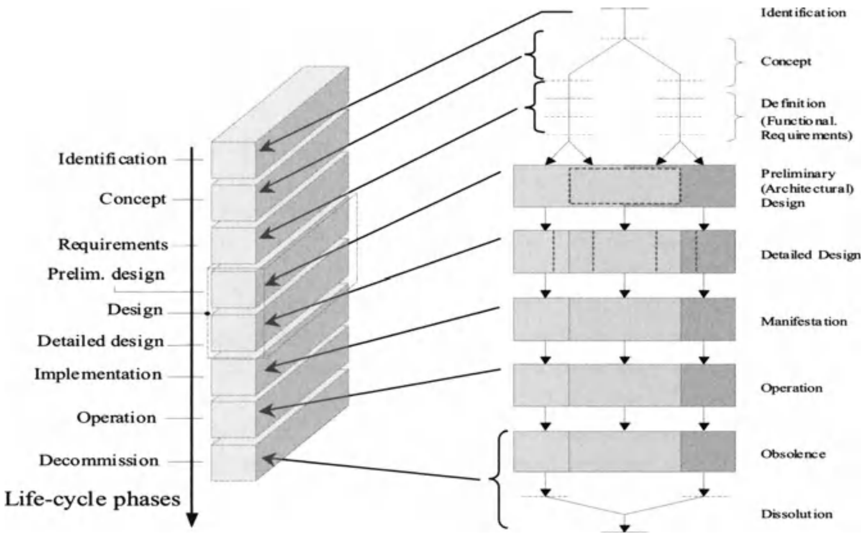


Fig. 3.2. Mapping of the PERA life cycle phases to GERA [22].

3.2.2 Life Cycle Phases in GRAI-GIM

The GRAI Integrated Methodology (GIM) adopts a "structured approach" (Doumeingts et al. , 1992), representing a flattened[23] version of the IMPACS[24] life cycle phases and its modelling methodology. The life cycle phases contained in GIM may be used for mapping against GERA as shown in Fig. 3.3. A brief justification of this mapping follows.

The Initialisation phase (overall definition of the company) corresponds to the Identification phase of GERA, while the Definition of the Domain of the study[25] matches the Concept phase of GERA.

The Analysis phase in GRAI-GIM produces the conceptual model (the 'what') and it is the equivalent of the GERA Functional Requirements phase. The end of the GRAI Analysis phase produces 'consolidated user requirements'. This is consistent with the result of a (functional) requirements analysis phase in GERA.

In the GRAI sense, the design phase (at the structural level) consists of user-oriented- and technically-oriented sub-phases. The user-oriented design sub-phase intends to obtain user validation and support for the design, while the

---

[22] only life cycle phases have been represented - other aspects omitted for clarity.

[23] due to the omission of the Abstraction dimension.

[24] Integrated Manufacturing Planning and Control System (IMPACS) framework (Chen et al. , 1990) - refer Fig. 3.19 for a graphical representation.

[25] although part of the of the Initialisation phase (Chen and Doumeingts , 1996), Domain Definition may be mapped onto the separate phase of Concept in GERA.

technically-oriented sub-phase contains the transformation of the user specifications into a set of technical specifications. This transformation involves a change in the number and nature of the modelled aspects[26]. The transformation of user requirements into user / technical specifications is captured in a single GERA life cycle phase, which is the Preliminary Design[27]. Preliminary design is often called Architectural Design, because it is the phase where requirements are translated into structures that satisfy the requirements.
The Implementation phase (at the Implementable level) is intended to present possible (and likely) choices for the technically oriented specifications produced in the technically-oriented design phase of GIM.

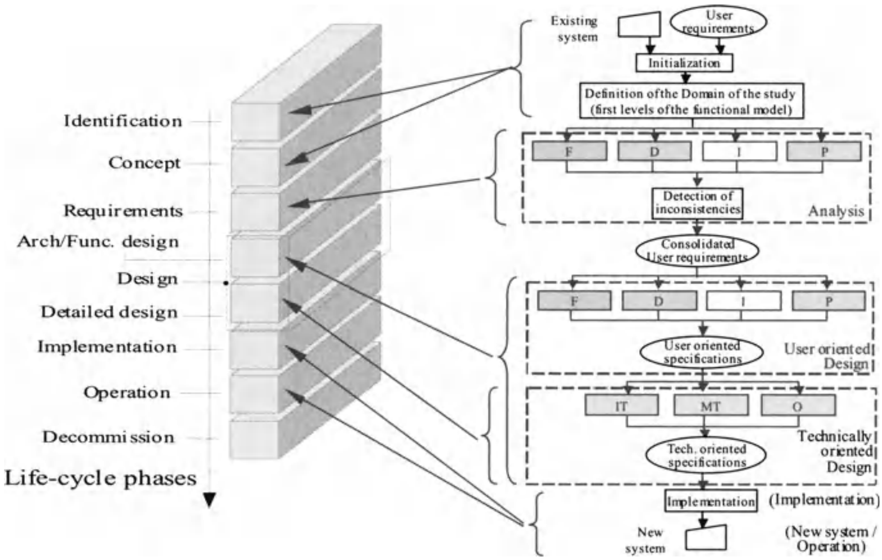


Fig. 3.3. Mapping of the GRAI-GIM structured approach steps to life cycle phases of GERA (Bernus et al. , 1996b)[28]

Therefore this phase maps onto the detailed design phase of GERA (which carries on the same task, with the same outcomes).

[26] from four ***views*** (function, information, decision, physical) to three ***domains*** (information technology, manufacturing technology and organisation)

[27] Preliminary Design is an essential stage (recognized as such in all reviewed architectures), in which system requirements are derived from the user requirements. This phase should involve people (architect(s)) that understand both the business (the users) and the system (the artefact being built to satisfy the user needs) and is thus able to bridge the two realms.

[28] The GRAI methodology model ('structured approach') has been used here for the purpose of life cycle phases' mapping. The GRAI modelling framework will be used for the framework mapping in Section 3.4.2.

The original GRAI model[29] was conceived in order to respond to a perceived lack of emphasis in CIM on Production Management. Nowadays, the targeted domain of the GRAI-GIM modelling framework (and the associated Structured Approach) is still by and large the support of the design of CIM systems (and more recently of other types of enterprises) by producing accurate technically-oriented specifications. As such, GRAI-GIM does not explicitly provide modelling constructs for the Implementation, Operation or Decommissioning life cycle phases as represented in GERA[30].
Other important aspects regarding the mapping of the GRAI framework to GERA are discussed in Section 3.4.2.

3.2.3 CIMOSA Life Cycle Phases

The Computer Integrated Manufacturing Open System Architecture framework (CIMOSA[31]) displays life cycle phases which may be mapped against GERA's phases in a relatively straightforward manner (refer Fig. 3.4). One of the main purposes of CIMOSA is to produce a formal, executable model that may ultimately be used to simulate and operate[32] the enterprise. As such, CIMOSA concentrates on the user requirements gathering, preliminary and detailed design life cycle phases. CIMOSA assumes that the need for change or for a specific artefact has already been identified (Identification phase in GERA). It also assumes that the decision has been taken to react to this need in a specific way (i.e. the GERA Concept has been formulated). The Requirements Definition of CIMOSA aims at user requirements gathering and as such maps onto the Requirements life cycle phase of GERA. The Design Specification phase produces the system specific / functional requirements from the user requirements - and therefore it maps onto the Preliminary Design phase of GERA. The Implementation Description phase plays multiple roles in the CIMOSA life cycle, i.e.:

- a detailed enterprise model is produced;
- the executable model is developed and implemented;
- the model is simulated, evaluated and validated.

Therefore, the Implementation Description phase of CIMOSA may be mapped on both the Detailed Design and the Implementation phases of GERA depending on the domain and enterprise modelled[33].

[29] initially resembling a collection of methodologies with no explicit framework.
[30] The GRAI Methodology (GIM) does include however descriptions of the Implementation phase and therefore it would include processes that produce deliverables (such as business plans) to cover this life cycle phase.
[31] CIMOSA (www.cimosa.de) has been the basis for the ENV40003 standard.
[32] i.e. through model-based control.
[33] this Implementation Description mapping is e.g. typical of software engineering activities. In software production, coding (the creation of an executable software) involves most detailed design decisions.

CIMOSA emphasizes the necessity of a formal 'Release to Operation' phase, which transfers the executable model from the enterprise engineering environment to the operational environment. The use of two separate environments (one for model engineering and another for model execution) supports CIMOSA's interpretation of parallel and concurrent processes[34]. Therefore, the Release to Operation phase in CIMOSA may be mapped to the GERA Implementation phase.
The released (runnable) model is then executed by the Services of the CIMOSA Integrated Infrastructure (IIS) in the Execution phase of CIMOSA, which hence maps to the Operation phase of GERA. The Change and Maintenance phase illustrated in (Kosanke and Vernadat , 1998) is understood in Fig. 3.4 as encompassing all reiterated life cycle phases involved in a specific change process (occurring in the engineering environment and in parallel with the normal operation of the enterprise)[35].

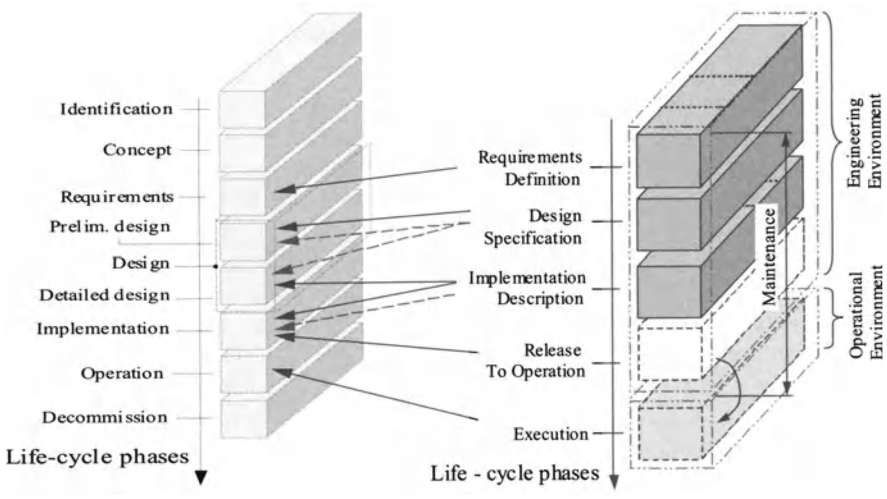


Fig. 3.4. Mapping of CIMOSA life cycle phases to those of GERA[36].

[34] more about these concepts in Section 3.3.4.

[35] it is therefore justified to consider CIMOSA's Change and Maintenance phase as the concept corresponding to a Change and Maintenance stage in the entity's life history.

[36] all other dimensions except for the life cycle have been omitted for clarity. Release to Operation, Execution and Maintenance are not explicitly included in the basic CIMOSA reference architecture but are defined within the CIMOSA framework. **NB** The dashed arrows represent the changes in the meaning of life cycle phases expected in the *new* CIMOSA baseline, as subsequently explained in the main text.

CIMOSA does not explicitly contain any life cycle phase(s) dealing with retiring the enterprise entity at the end of its useful life that could be mapped against the Decommission phase of GERA.
NB It is expected that possibly in the new CIMOSA baseline[37] the meaning of several life cycle phases may change as follows (refer the dashed arrows in Fig. 3.4):

- CIMOSA Design Specification will map onto both GERA Preliminary and Detailed Design phases;
- CIMOSA Implementation Description will map onto the GERA Implementation (Build) life cycle phase.

3.2.4 The Zachman Life Cycle Aspect

The Zachman framework takes a somewhat different approach towards life cycle, presenting life cycle phases as perspectives of the various stakeholders involved in the enterprise engineering effort. The concepts of life cycle and life cycle phases are not explicitly present, however a mapping is possible. This is because different stakeholders use different levels of abstractions to consider the enterprise entity in question, and these abstractions correspond to GERAM life-cycle phases. **NB** A direct mapping is not possible since GERA contains types of activities, while Zachman describes *deliverables* that given stakeholders produce.
The Zachman framework does not include a stakeholder perspective that can explicitly and completely match the GERA Identification phase. However, in analyzing the contents of the Zachman cells in the Objectives / Scope row (refer Fig. 3.38) one realizes that these cells contain deliverables that describe the business[38](very high-level functional / organisational structure, boundaries, etc - which match deliverables created in the GERA Identification phase) but also business goals and strategy (which would be developed in the GERA Concept phase). Hence, the Objectives Scope perspective maps[39] onto the GERA Identification and Concept life cycle phases.
As shown above, a single GERA phase may map onto several Zachman perspectives and several GERA phases may converge to a single Zachman perspective. As an example: the Requirements phase maps to both the Business Owner's perspective and the Architect's perspective. The justification for this is that the architect should be able to gather user requirements and translate them into system requirements (which must be satisfied by the Preliminary Design). The architect's bridging across user- and system requirements

[37] and EN/IS 19439 superseding the ENV40003, which was based heavily on CIMOSA
[38] in the following 'business' and 'enterprise' will be interchangeably used.
[39] although a direct mapping is not possible, the term 'mapping' is used in this section with the meaning of association of *deliverables* and *stakelholders perspectives* (Zachman) with *types of activities within a life cycle phase* (GERA).

is optimal if the architect is involved both in the GERA Requirements and Preliminary Design phases.

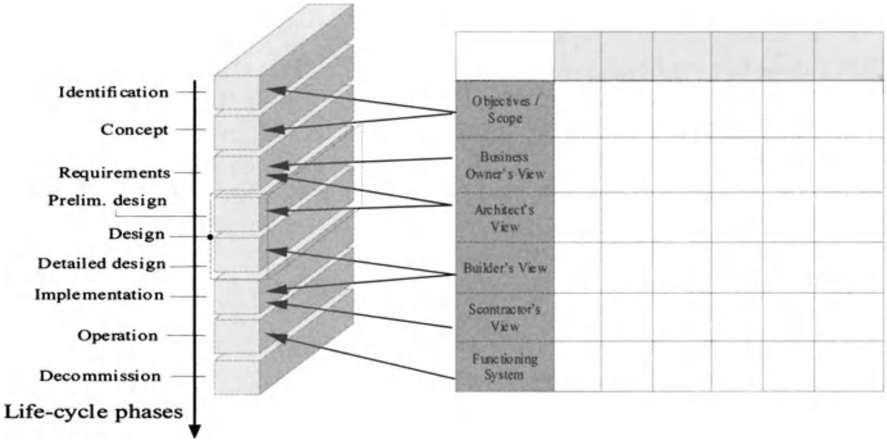


Fig. 3.5. Mapping of Zachman perspectives to life cycle phases of GERA[40]

NB Such 'mappings' depend on the scope and size of the artefact being modelled and consequently on the modelling strategy adopted. For example, in the case of a large project the 'builder' may be the Technology Designer, accomplishing the detailed design[41] but also being involved in the Implementation phase. The subcontractor may however only be involved in the Implementation phase. The meanings attached to the Zachman framework cell designations by the users also vary considerably (e.g. depending on the user's background and current involvements) and they may affect any potential mappings[42].
Users of the Zachman Framework may find it useful to add another column to the Zachman framework, labeling it with GERA life cycle phases (potentially with even finer grain subdivision of phases than used by GERA), and then assigning concrete human roles to these (according to the problem at hand) as per the first column of the original Zachman Framework. Thus the practitioner has a map of 'who does what'. This would also mean that the perspective of one human role (e.g. the architect) would include more then one model (such as a functional requirements model and an architectural model) if necessary.

[40] other Zachman and GERA frameworks' dimensions omtted for clarity.
[41] i.e. allowing for constraints such as tool / technology/ material availability, relevant standards / laws, etc (Zachman , 1987)
[42] e.g., refer the (Hay , 2000) mapping from the perspective of the Oracle life cycle model.

In conclusion, the Zachman framework represents the perspectives of the human roles[43] involved in the life cycle phases described by GERA.

3.2.5 Life Cycle in C4ISR

C4ISR does not explicitly contain a dedicated life cycle concept. It does, however, contain three main architecture 'views' which are not directly supported by existing systems engineering formalisms and tools (Levis and Wagenhals , 2000). A close review of the C4ISR document however leads to a possible implicit mapping of the C4ISR views onto the GERA life cycle phases (refer Fig. 3.6). As a general observation, the C4ISR views contain deliverables that address multiple GERA life cycle phases.
The C4ISR Operational view plays two major roles:

- concise description of the action(s) taken to respond to an identified[44] need for an artefact or its change;
- details of the previous description ('what needs to be done' in terms of the end user).

Within the Operational view, the architect elicits and structures the needs of the end user. Therefore the best match[45] of the Operational view are the Concept and Requirements life cycle phases of GERA.
The C4ISR Systems view identifies capabilities and characteristics to fulfill the requirements gathered in the Operational view[46]. Thus the main scope of the Systems view corresponds to the Preliminary Design phase in GERA.
The C4ISR documentation however, also describes the Systems view as identifying specific enabling technologies and detailing the design as necessary. Preliminary Design as understood in GERA should not be linked to any particular technology and be limited to functional requirements, in order to preserve its neutrality (and therefore reusability potential) in respect to changing enabling technologies. Therefore, the Systems view should also map onto the Detailed Design life cycle phase of GERA. In conclusion, the Systems view maps in principle[47] onto both the Preliminary Design- and Detailed Design phases of GERA. The Systems view thus includes the preliminary design of the artefact as well as any descriptions (on the preliminary or detailed design

[43] it is possible for example to identify the Objectives/ Scope view with a 'Manager's (or CEO's) View' and the 'Functioning System' view with an 'Operator's View'.
[44] the C4ISR architecture does not explicitly cover the identification of the need for an artefact, or the need for change.
[45] as previously shown, the C4ISR views do not explicitly follow the commonly accepted life cycle paradigm (containing a set of well-understood life cycle phases).
[46] in a building industry analogy, the architect must elicit the system constraints out of- and matching the user requirements.
[47] in principle, since the (non-essential for a specific purpose) deliverables identifying specific technologies may be simply omitted from the System view. In this case, the System view would map solely onto the GERA Preliminary Design phase.

level) that *constrain* this process. (In GERAM these constraints are expressed as a set of reference models used by the preliminary design activities.)
In the Detailed design process, the designer seeks to satisfy the system (functional) requirements (contained in C4ISR's System view) derived from the user requirements (contained in the C4ISR Operational view), by taking into account the relevant constraints: legal (standards and other regulations), physical (laws of physics, materials, etc) and technological. C4ISR identifies the Technical view of the architecture as being "the set of rules that governs system implementation and operation" (C4ISR Architectures Working Group , 1997). Therefore the C4ISR Technical view maps onto (part of) the Detailed Design phase of GERA.

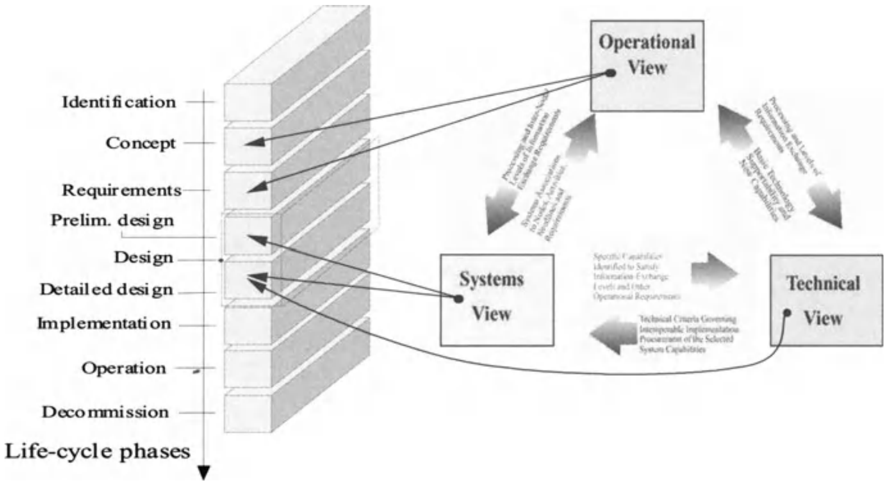


Fig. 3.6. Mapping C4ISR views to life cycle phases of GERA

As a conclusion, the Operational view contains '*what needs to be done*' both in concise (concept) form and in user requirements terms. The Systems view contains the architectural design and the part of detailed design referring to the enabling technologies used (physical / technological constraints). Finally, the Technical view contains the rest of detailed design, i.e. the applicable standards and regulations (legal / administrative constraints).
The case of GRAI-GIM (Section 3.2.2) has shown that some architecture frameworks may contain implicit life cycle aspects in more than just one component (e.g. in GRAI-GIM the life cycle aspect exists both in its modelling framework (reference architecture) *and* the GRAI structured approach).
It has therefore been considered worthwhile to attempt a mapping using the C4ISR-supplied high-level architecture engineering guidance (refer Sec-

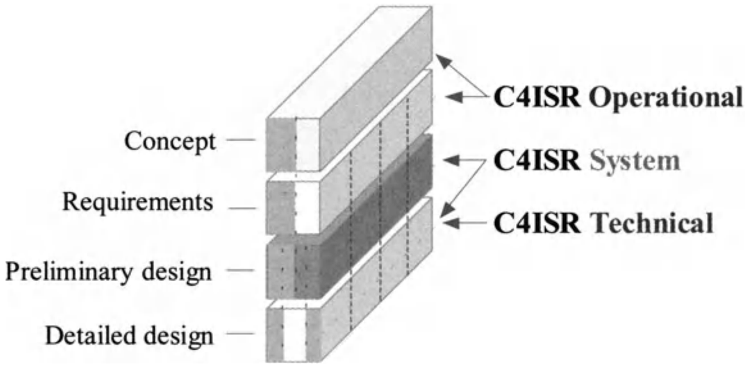


Fig. 3.7. Extract from Fig. 3.6

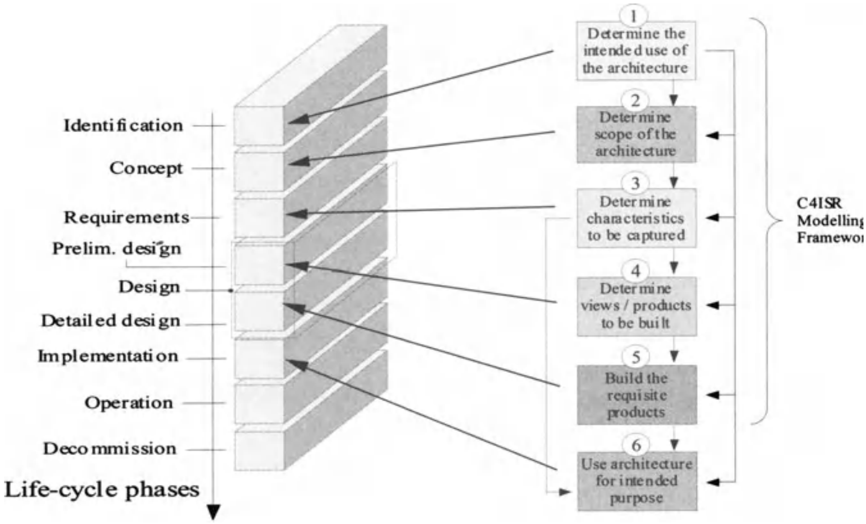


Fig. 3.8. Mapping of the six-step C4ISR process onto the GERA life cycle phases.

tion 3.6.5 and Fig. 3.44 for details). The result of this mapping is shown in Fig. 3.8 [48].

[48] The feasibility of such a mapping is consistent with the view that a life-cycle architecture may be considered an abstraction of an enterprise engineering methodology.

3.2.6 Life Cycle in ARIS

The ARchitecture for Information Systems (ARIS) defines the concept of life cycle 'levels' or 'phases' explicitly. However, these phases are claimed to be descriptive[49] in nature, rather than components of a procedural model (e.g. steps) towards building an information system. At a closer analysis however, the ARIS life cycle phases are very similar to those of CIMOSA (refer Section 3.2.3 and Fig. 3.4). In comparison to CIMOSA though, ARIS identifies further two phases, namely the Operational Business Problem and the Information Technology levels. A brief description and justification of the ARIS life cycle phases and mapping onto the GERA follows.

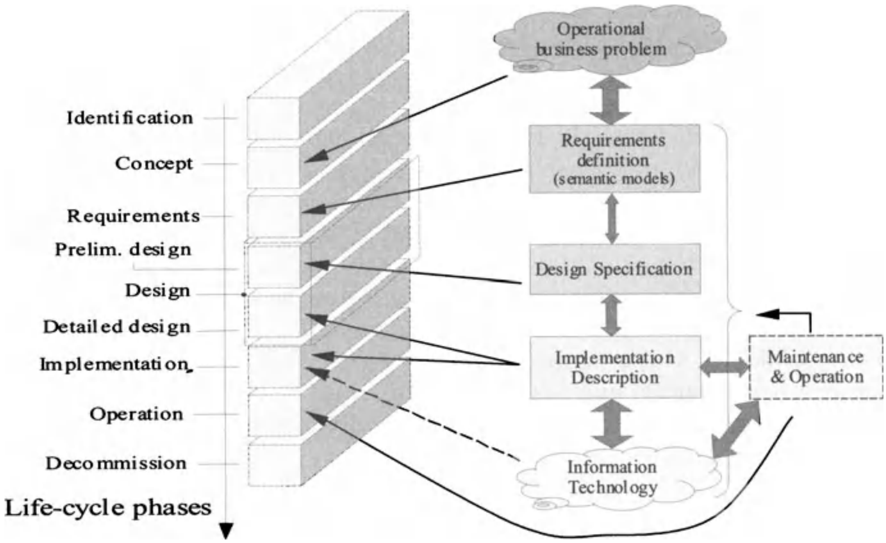


Fig. 3.9. Mapping of the ARIS levels onto the GERA life cycle phases (based on (Scheer , 1999))[50]

The Operational Business problem level contains high-level descriptions of the solutions proposed to the needs that are considered to have already been identified[51]. These descriptions are constructed using semi-formal descriptive methods. Therefore this level maps onto the GERA Concept phase.

---

[49] describing the proximity of the level in question relative to the Implementation phase (Scheer , 1999).

[50] the width of the double arrows used between the ARIS levels (right-hand side of picture) depicts the degree of coupling between levels - the wider the arrow, the tighter the coupling.

[51] similar to CIMOSA, ARIS does not explicitly distinguish an Identification life cycle phase.

The Requirements Definition level has to further develop the semi-formally defined concepts in the Operational Business Problem level using a more formal language. This is done for the purpose of using these concepts for subsequent system development. The purpose of (and the tasks present in-) the ARIS Requirements Definition level suggest that this level may be mapped onto the GERA Requirements level.
The Design Specification level as described in ARIS translates the user requirements gathered in the Operational Business Problem level and further refined in the Requirements definition level into 'general information technology interfaces'. i.e. system requirements. Therefore the ARIS Design Specification Level maps onto the GERA Preliminary Design life cycle phase.
The Implementation Description level "assigns the design specifications to concrete hardware and software components" (Scheer , 1998a), therefore matching the GERA Detailed Design and partly the Implementation life cycle phases.
Finally, the Information Technology level intends to depict the real system and as such it maps onto the Implementation life cycle phase of GERA.
An Operation and Maintenance phase has been subsequently been added to the ARIS life cycle (Scheer , 1999).The Maintenance aspect is similar to that of CIMOSA (covering a slightly different set of life cycle phases), acknowledging the necessity to maintain the accuracy of the model via subsequent update cycles (and thus also reflecting a temporal aspect in ARIS). The ARIS Operation component of this phase maps onto the GERA Operation life cycle phase, while the Maintenance component covers the set of life cycle phases between Requirements and Implementation.
ARIS does not explicitly cover entity obsolence (decommissioning) aspects in the GERA sense.

3.2.7 Conclusion About the Life Cycle Concept

The life cycle concept as understood in GERA is (implicitly or explicitly) present in all reviewed architecture frameworks, however not necessarily within their modelling framework component[52].
Any voluntary creation or change of an entity by humans is a result of the human perception of either the need for that (type of) entity or for the entity's change. These perceptions may be ideas, aspirations and / or visions of what could be created or changed. This pool of perceptions may be contained in an important, but informal separate life cycle phase. GERA covers this type of phase in its Identification; however, it does not provide a full set of view subdivisions, because of the informal and varied character of this identification activity. The Identification phase may be accompanied by another phase (e.g.

[52] life cycle aspects may commonly be present within e.g. the *methodology* component of the architecture framework in question.

called Concept in GERA) in which decisions are made as to *how* to react to the Identified need for an entity or its change[53].
Any entity being conceived, designed, implemented and possibly executed will eventually reach the end of its useful life. Many of the reviewed life cycle architectures do not explicitly cover this aspect, although this is the very phase when (a) a decision is made to either retire or revamp the entity and (b) irrespective of which decision is made in (a), the relevant enterprise knowledge and other resources must be properly transformed[54], either for their preservation or their reuse.
A 'complete' (in GERA sense) reference architecture should attempt to also cover these two life cycle phases to a depth adequate to the purpose of the modelling task.

3.3 Life History: The Timeline Aspect in Methodologies

Any business entity[55] aspiring to a lasting presence in today's ever changing environment must *adapt* to its surroundings, via change processes of its own. An agile[56] enterprise not only copes with, but also thrives on (and even preempts) the environment changes. For that to occur, the change process as a whole must be practically continuous.
Most of the change processes occurring during the life of an enterprise are concurrent and interacting with one another. This character of change processes within an enterprise may be suitably modelled using the GERA *life history* concept[57].
The concept of life *history* has often been mistakenly identified with that of life *cycle*. GERAM makes a difference between life history - "the actual sequence of steps a system *has* gone (or *will* go)[58] through during its lifetime" and life cycle - seen as "the finite set of generic phases and steps a system *may* go through over its entire life history" (ISO/TC184/SC5/WG1 , 2000b).
Each reviewed architecture covers the life history concept differently. Most of them do not have an explicit life history concept, and only implicit mapping

[53] some of the ideas, aspirations, etc in the Identification phase may simply be dismissed or forgotten; the content that *does* filter into the Concept phase will however trigger a response, e.g. to react to the need for change by a change project, or to temporarily postpone a response to that need.
[54] e.g. non-human resources / knowledge must be properly archived and human resources retrenched/retired etc in the case of obsoletion. In the case of artefact revamping, existing non-human resources may need converting / aggregating / refining etc, while existing human resources may need retraining.
[55] hereafter also referred to as an "enterprise" or "enterprise entity"
[56] agility: "capability to respond" (Goranson , 1998).
[57] note that an important part of these change processes involves the life histories of *other* enterprise entities.
[58] author's comment.

is possible. The completeness and adequacy of this coverage may be assessed in two stages:

- understand the GERA life history concept via a non-trivial example;
- map any explicit or related/usable constructs found in the reviewed architectures to the GERAM concept.

For a typical life history relevant to networks of virtual enterprises refer Chapter 7 in this Book.

3.3.1 The Life History Aspect in GERA

The following builds on the simple example contained in Annex A of ISO/IS 15704 (ISO/TC184/SC5/WG1 , 2000a).
This example illustrates a way of representing part of the life history of an enterprise entity, relevant to a particular change process - which is managed via separate change and construction projects. Also represented are the applicable life history sections and views of the entity *types*[59] involved in the process.
Consider the following scenario. The management of an enterprise identifies a need for change that would affect the 'design' of the company[60]. Due to the specifics of the change, the decision is made to involve both in-house resources (from management and from the operations) and external resources (consulting companies, construction enterprises[61]).
The proposed change process is part of the life history of the enterprise - therefore it has a time dimension and it includes life cycle phases in a particular sequence. To minimize any disruption to the enterprise, the change process would typically run in parallel with the normal operational processes of the enterprise, except for the short period when the changes are implemented. Such concurrent / parallel processes may be represented in a set of diagrams such as those shown in Fig. 3.10, showing life cycle phases vs. time for each entity involved.
Each diagram also shows a view of management and control vs. service to the customer and production (operational) processes, detailing the interaction and

[59] e.g. enterprises, projects, products, etc. NB these entities may simultaneously play different roles: a manufacturing enterprise entity may also be the product entity of another manufacturing enterprise entity; this also implies a possible recursion of the relations between enterprises. Refer Section 3.8.2, Chapter 2 and (ISO/TC184/SC5/WG1 , 2000a) for details.

[60] i.e. the way the company operates. Remember that an agile enterprise should be able to do this promptly and quite often (in fact continually). Please note that some changes may be handled by the (agile or not) enterprise in its current design, i.e. as part of its usual operation - such as e.g. rescheduling of production, etc.

[61] the terms 'consulting' and 'construction' may refer here to *any* domain (including e.g. software development - in which case the term 'construction' would refer to software engineering).

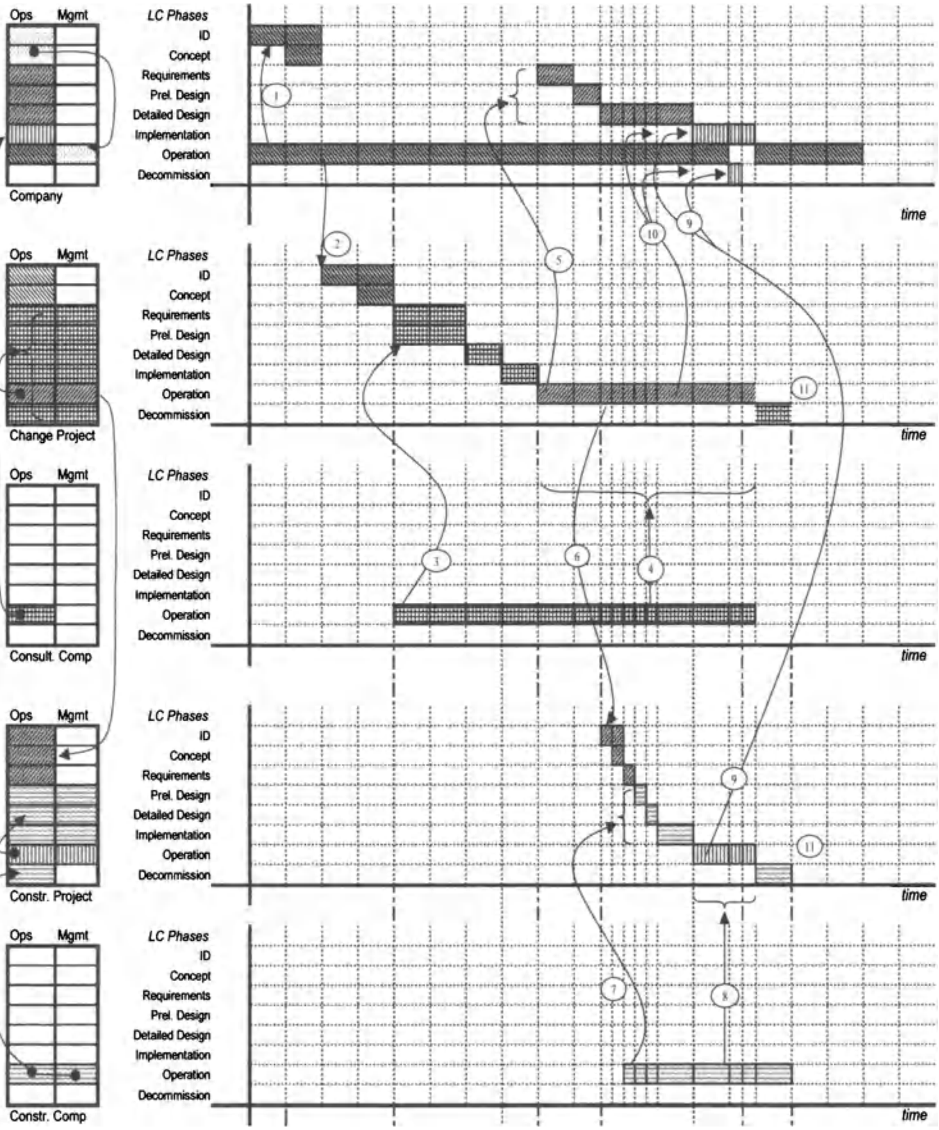


Fig. 3.10. Life Cycle - Life History relation and enterprise entity type recursion according to GERA (ISO/TC184/SC5/WG1 , 2000a), for a typical change process.

involvement of each entity type in the change process[62]. On the timeline, it is possible to aggregate parts of the change process into *stages*, with each stage typically ending in a milestone, or decision point.
Explanation of Fig. 3.10:

- The management of the enterprise identifies the need for change and defines the concepts underlying the change (i.e. the need to act and in what way - e.g. via setting up a change project). This is shown in stages **1** and **2**;
- The consulting company is called in to assist with setting up and managing the change project. Therefore it designs the change project (stage **3**) and manages it when the project starts operating (stage **4**);
- The change project conceived by the enterprise's management is (in this example) designed and managed by the consulting enterprise. When the project starts operating:
  - it defines the requirements, the preliminary and the detailed designs of the necessary changes (stage **5**);
  - it sets up[63] the construction project (stage **6**);
  - it supervises the implementation of the changes once the construction project starts operating (stage **10**).
- The construction project is derived from the change project (stage 6), designed (stage **7**) and managed (stage **8**) by the construction entity. This project accomplishes the actual implementation of the changes and any necessary obsoletion of existing parts of the enterprise entity in question (stage **9**);
- Once the changes are implemented, both the change project and the construction project are decommissioned (stage **11**).[64].

Besides illustrating the life history, the example presented in Fig. 3.10 demonstrates another important concept: the relation between enterprise entities. This relation is explained in ISO/IS 15704 using the concept of *recursion* [65] in (ISO/TC184/SC5/WG1 , 2000a); the same concept also exists (under various designations) in several architectures (e.g. Zachman framework). Recursion is explained in more detail in Section 3.8.2.
NB A very important special case of multiple concurrent life cycle activities is *Concurrent Engineering, which* is an approach to product design. Concurrent

[62] in GERA (ISO/TC184/SC5/WG1 , 2000a) this view is called the *entity purpose* view.
[63] up to the requirements level only; the construction company accomplishes the design.
[64] at this *stage* it is vital that the knowledge accumulated is properly preserved for future reuse (e.g. in the form of a partial model) - i.e. all necessary decommissioning phase activities are carried out.
[65] recursion on e.g. the support of an entity type's various life cycle phases by a finite set of entity types, some of which may be in turn supported by entity types from the same set. Refer Chapter 2 and (ISO/TC184/SC5/WG1 , 2000a) for details.

engineering advocates the simultaneous design of products and of the production facilities and processes to produce them. Adoption of this approach may significantly reduce the cost- and time needed to develop and produce new products and it is widely used e.g. in the automotive industry. Since simultaneous engineering requires frequent interactions between the life cycle processes of the product- and production entities, specialised tools and organisational designs have been developed to support model exchange, feasibility- / consistency- / cost analysis and collaborative design.

3.3.2 PERA and Life History

The PERA documentation refers to the concept of life history explicitly. However, the PERA documentation appears to use the concepts of life history and life cycle interchangeably - e.g. when referring to the phases represented in the PERA reference architecture (Williams et al. , 2001).
This could cause some confusion regarding a possible temporal aspect associated with the PERA life cycle phases[66].
A clear temporal aspect however is included in the PERA Handbook for Master Planning in describing the tasks involved in producing AS-IS and TO-BE architectures. This also includes a transition path and training plan, which enable the change process(es) involved in the evolution from the AS-IS to the TO-BE states (refer Fig. 3.11).
Other temporal aspects include iterations and parallel enterprise integration activities, as detailed in (Williams et al. , 1998a). Similar to CIMOSA and GERA, the best way to illustrate the life history concept is to represent time on a separate axis.

[66] in the unambiguous interpretation proposed by Noran (2000a), life history is composed of *stages* uniquely defined in time. Each stage comprises a life cycle *phase*. Life cycle phases may appear in one- or several life history stages (or milestone).

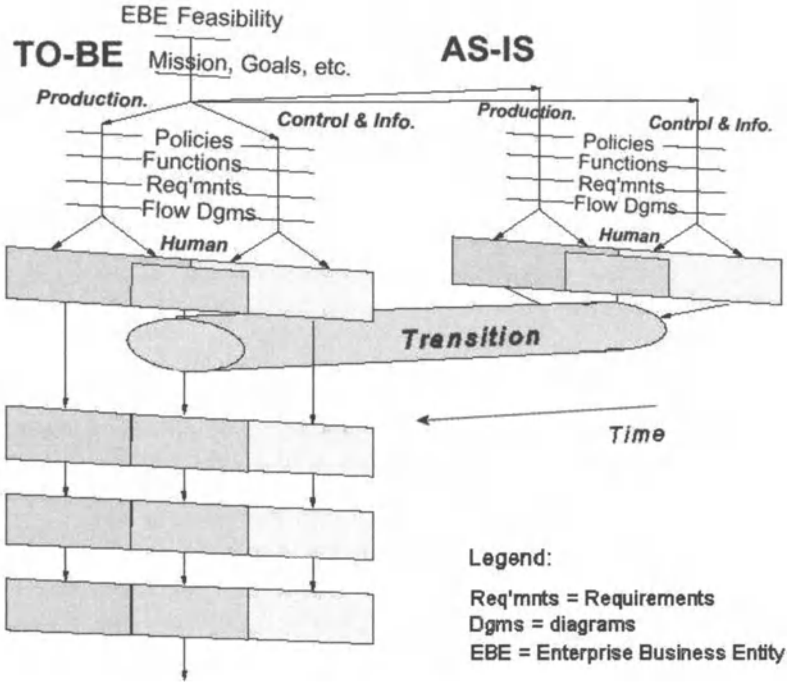


Fig. 3.11. Temporal aspects as representd in the PERA Handbook for Master Planning (based on (Williams et al. , 2001))

Figure 3.12 represents a typical example of a life history. where the need for e.g. additional preliminary design work affects other life cycle phases.
The figure also illustrates the concept of *parallel* and concurrent processes (since some of the original life cycle phases may still happen in parallel with the preliminary design phase's reiteration).

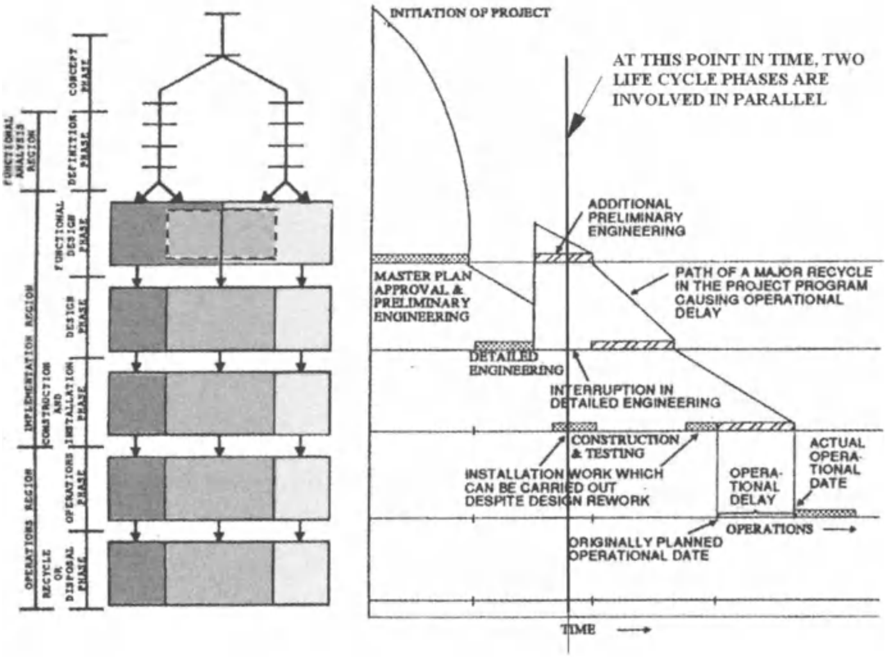


Fig. 3.12. Life cycle - life history relation example in PERA (based on (Williams et al. , 1998a))

3.3.3 Life History Aspects in GRAI-GIM

Life history is not explicitly defined in GRAI-GIM. However, temporal aspects are present in the application of the GRAI-GIM structured approach - e.g. the Structural Design phase of the structured approach takes into consideration the evolution of the system. GRAI also includes a methodology called the GRAI Evolution Methodology (GEM, (Doumeingts , 1998a)) that aims to provide a structured approach towards managing transition between AS-IS and SHOULD-BE models of the entity in question. GEM acknowledges the changing nature of the modelled entity and its environment. Consequently, GEM advocates the *continuous adaptation* of the SHOULD-BE model. On a shorter time span[67], a *reachable* state (which GEM calls NEXT STEP) may be defined.

GRAI-GIM also contains temporal aspects in its other methodologies. Typically, these aspects are contained in project timelines or GANTT type charts,

[67] in GRAI-GIM terminology, the NEXT STEP is reached over a strategic *period*, while the SHOULD-BE should in theory be reached over a strategic *horizon* - where period < horizon. The classic example of period vs. horizon is illustrated in the GRAI Grid (Fig. 3.49)

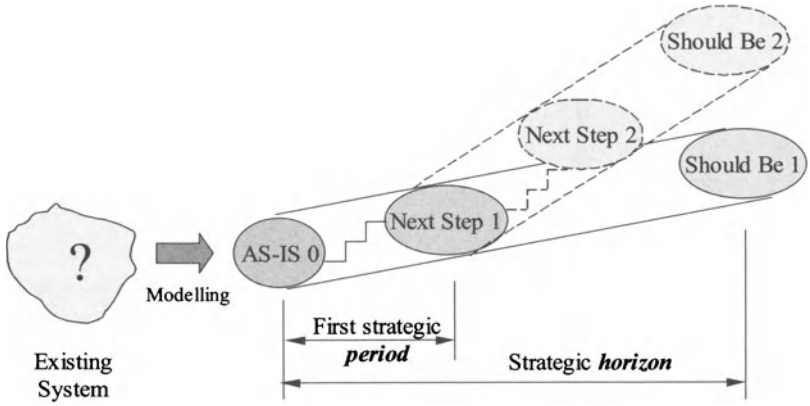


Fig. 3.13. Evolution process in the GRAI Evolution methodology (adapted from (Doumeingts , 1998a))

describing at a very high level the milestones in the application of the methodology in question (e.g. for the Structured Approach or performance evaluation for EcoGRAI, etc). Such timelines / charts may be enriched with the real life cycle phases involved up to each milestone, represented orthogonally to the timeline (similar to GERA - refer Fig. 3.10).

In conclusion, while there is no explicit reference to life history in the GRAI documents reviewed, GRAI contains several implicit timeline aspects. Furthermore, there is no obstacle in using the GERA life history concept in connection with GRAI-GIM.

3.3.4 Life History Representation in CIMOSA

CIMOSA does not include an explicit life history concept. However, a life history may be represented using the GERA life history concept, similar to Fig. 3.10. Such a representation is shown in Fig. 3.14, which also illustrates the possible use of the CIMOSA change / maintenance processes to model life history. Maintenance may span (involve) one or several life cycle phase activities, depending on the scope of the particular change process.

Notes about Fig. 3.14:

- the parallel processes occur in separate environments (as shown);
- further life cycles (e.g. of the products of the modelled entity) may be defined in the Operating Environment;
- in the CIMOSA view, the released model should be executable (see below) and therefore little or no disruption of the operation phase may occur on the changeover from the old to the new executable model. This is reflected in the narrow gaps on the timeline in the Execution life cycle phase.

- the CIMOSA-defined 'Change' phase is represented here as a set of life cycle processes involved in that particular change except the normal operational processes.

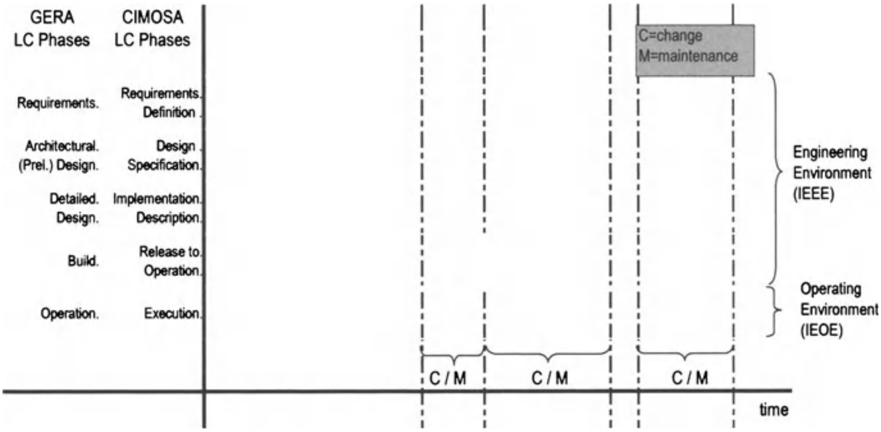


Fig. 3.14. Possible representation of the Life Cycle - Life History relation in CIMOSA

CIMOSA includes several other aspects relevant to life history. For example, CIMOSA allows for the creation of 'AS-IS' and 'TO-BE' models[68], which contain an inherent temporal aspect (since they represent the current and future (desired) states of the analysed domain).
Parallel and iterative processes are also recognised. CIMOSA considers enterprise integration to be a continuous process, which requires that the (possibly iterative) enterprise modelling activities should occur in parallel with the normal operation of the enterprise. CIMOSA has fulfilled this logical requirement by defining two separate environments in which the two types of processes may run independently: the Integrated Enterprise Engineering Environment (IEEE) and the Integrated Enterprise Operational Environment (IEOE). The IEEE allows particular implementation models to be produced in parallel with normal enterprise operation. The models are then released for operation and are immediately executable in the Operational Environment. This is possible because IEEE and IEOE share the same Integrating Infrastructure (IIS) Services, which are used for the modelling process and subsequently for executing the model (hence the consistency of the model is also essentially ensured).

[68] whereby the requirements and design of the 'TO-BE' model are obtained via abstraction from, and operation modification of the 'AS-IS' model (Vlietstra , 1996).

The concept of Change / Maintenance may be used to refer to all the necessary life cycle phases involved in updating the model contained in the Engineering Environment. Subsequently, in the Release to Operation phase one or several models may be specialised and released into the Operational Environment.

3.3.5 Life History Aspects in Zachman

The Zachman framework includes a concept that may be used to represent life history. Transition, an important construct in modelling life history is not explicitly present, although other aspects that could be used to model it do exist (Hay , 2000). Life history may be represented in the Zachman framework via the **When** column (as in Fig. 3.22 and Fig. 3.23), which implies temporality and succession. This may work well for successive / non-parallel change processes. Alternatively, one could simply omit the 'When' column from the Zachman Framework and represent it on a separate 'time' axis (as represented in Fig. 3.15 and omitted from mapping in Fig. 3.22)[69]. For each change process there may be separate deliverables (as accommodated in a Zachman Framework for that change project), while the occurrences of the life cycle activities for multiple change processes may be simultaneously shown on a combined time diagram. To preserve the applicability of tools that support the Zachman Framework, the temporal aspects of each separate change process could be combined on an aggregate time diagram, thus showing their possible interactions. Figure 3.15 shows a possible interpretation of life history in the Zachman framework, according to the GERA life history concept. In such a life history, there would be a purpose associated with each deliverable (characterised in the Zachman framework's 'Why' column). This purpose can refer to the *role* of the deliverables in the life history of the entity.
Another temporal aspect in the Zachman framework is the concept of versioning[70], described in (Sowa and Zachman , 1992) as one of the three aspects of recursion applicable to the Zachman framework[71].

---

[69] Conceptually, it is preferable to use a time axis that is orthogonal to other dimensions, because time is an independent dimension (e.g. processes (function), data, resources, locations (network), motivations may all change / evolve with time).

[70] versioning is a temporal concept since it identifies stages in the evolution (and therefore in the life history) of an artefact. **NB** versioning is present in an implicit form in all reviewed frameworks.

[71] refer Section 3.8.2.4 for details.

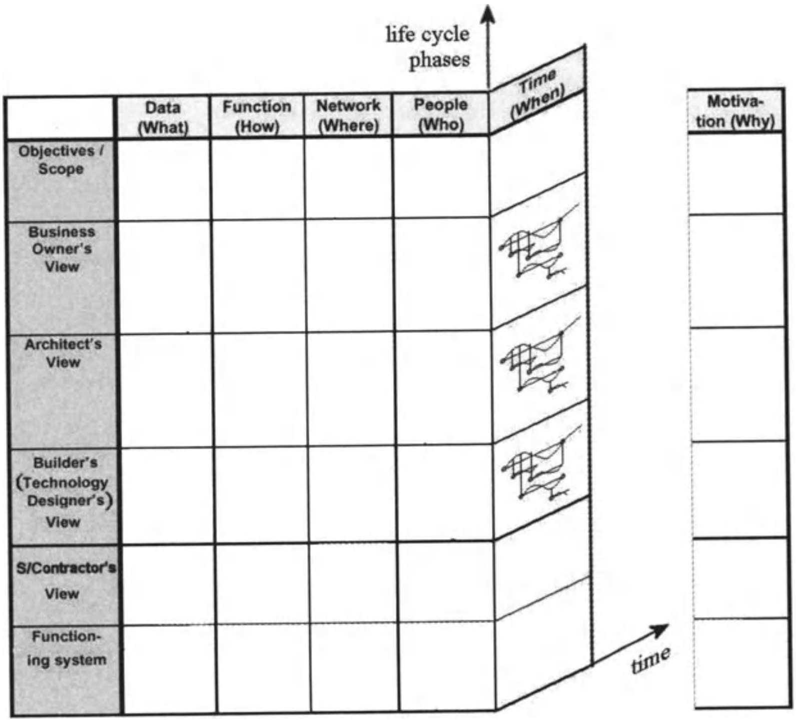


Fig. 3.15. Possible life history interpretation in Zachman

3.3.6 Temporality and Succession in C4ISR

Although C4ISR does not explicitlty define the concept of life history, temporal aspects are present. For example, C4ISR includes a System Evolution Description (SV-8)[72] which provides for both transition and migration type change processes[73]. However, the Evolution Description deliverable relies on several other views and only provides a high-level timeline comprising major milestones in the change process (similar to the GRAI project timelines - refer Section 3.3.3).

Figure 3.16 represents a possible extension of the System Evolution timeline with the views and products[74] specified in the C4ISR architecture framework.

[72] refer to Table 3.1 for a complete list of the C4ISR architectural products and their codes

[73] C4ISR's meaning of the terms *migration* ("incrementally creating a more streamlined, efficient, smaller and cheaper suite") and *evolution* ("spreading in scope while increasing functionality and flexibility") are arguably exceedingly specialised.

[74] the GERAM enterprise models (EMs) are called *architectural products* in C4ISR.

In such a representation, at any given point in time it is possible to determine the architectural views (life cycle phases of GERA) involved and architectural product being developed. In addition, at any given point in time, predictions and *what-if* scenarios may be developed in more detail.

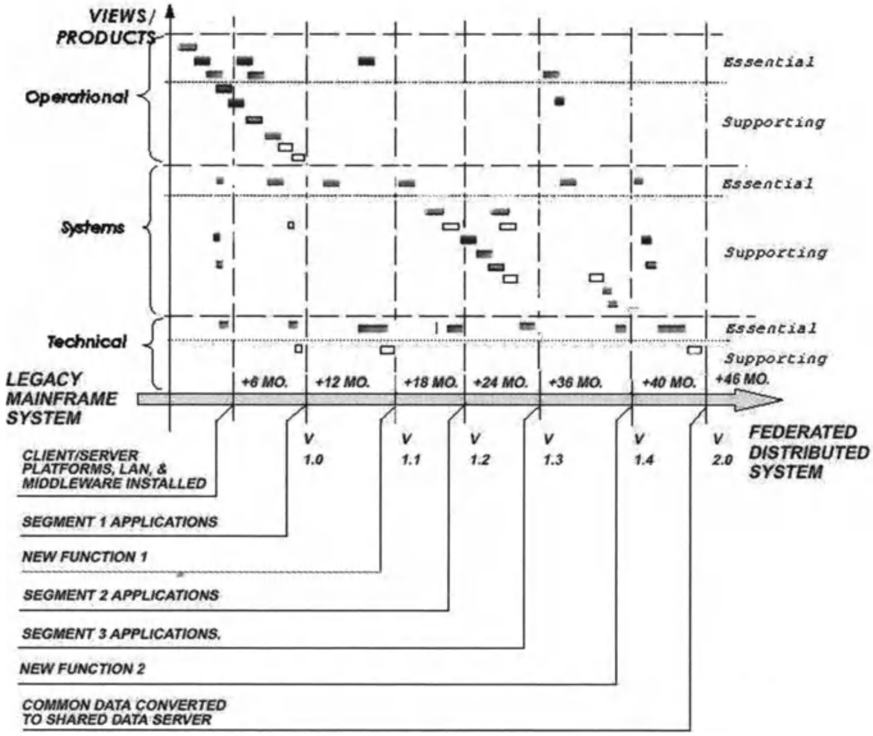


Fig. 3.16. Possible enriched System Evolution Diagram (SV-8) in C4ISR (based on (C4ISR Architectures Working Group , 1997)

The System Performance Parameters Matrix (SV-7), Systems Technology Forecasts (SV-9) and Standards Technology Forecasts (TV-2) also involve a temporal aspect, relevant to:

- the future life history of existing (designed and implemented) architectures[75];
- the future (to be conceived) architectures;
- any change processes (but especially of the transition type).

[75] here, with the meaning of 'systems architecture', or 'structure of the system';

Systems Technology Forecasts attempt to identify emerging technologies that may be employed at- and may impact on the preliminary design level. Standards Technology Forecasts endeavours to predict any future changes in standards applicable to the detailed design phase (which would impact on future architectures' compatibility and compliance[76]). The System Performance Matrix depicts relevant performance indicators for AS-IS and TO-BE states of the system. These indicators are geared to the Standards Technology Forecasts.

3.3.7 Life History Aspects in ARIS

ARIS does not define a dedicated life history concept. Temporal aspects are however present in its framework. For example, the ARIS-defined Operation / Maintenance life cycle phase provides for the necessary 'update cycles' (of various frequencies) for the ARIS levels, so as to keep the models consistent with reality. An orthogonal time dimension is, however, not represented explicitely. The representation of life history aspects may be achieved by users of the ARIS framework via applying the GERA life history concept as shown in Fig. 3.17. Just like in the previous cases (e.g. CIMOSA in Section 3.3.4), the most intuitive representation involves a separate representation of the life cycle phases vs. time. Note the increasing frequency of the update cycles when moving from the abstract to the more concrete content. **NB** A balance should be struck between the level of accuracy of the model and the resources that can be allocated for life cycle iteration (so as to produce that level of accuracy).
Other ARIS features containing temporal aspects include:

- simulation of the business processes (at process design level). Similar to all other frameworks that represent procedural process models, each step in a procedure can have an extent in time. Thus, process instances (as created in a simulation) can be unfolded in time;
- project / production scheduling (at process management level), allowing modelling of AS-IS and TO-BE states.

ARIS also covers versioning of the models created [77], which explicitly includes a temporal aspect (past models / alternatives, current model and possible future developments (Scheer , 1999).

---

[76] Systems- and Standards Technology forecasts are especially relevant in Software Systems Engineering where the pace of change (e.g. Information Technology) is significantly greater (and ever increasing) than in most other domains.

[77] ARIS supplies a metamodel for the version control.

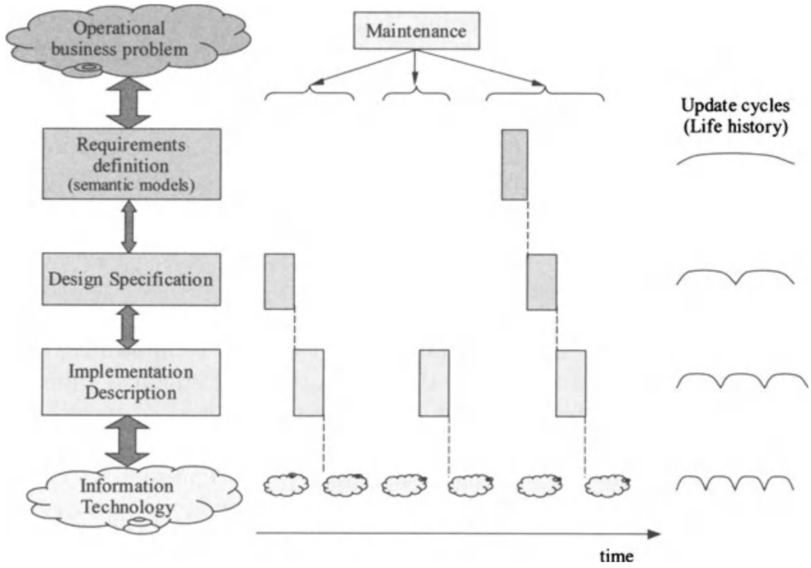


Fig. 3.17. Possible ARIS life history conceptual representation.

3.3.8 Conclusion About the Life History Concept

The concept of life history is important in modelling multiple and overlapping change processes. Acknowledgment of the concept of life history implies its separation from the life cycle concept and its consideration as an independent dimension. Life history is composed of life cycle phases (and implicitly, of the activities within these phases) that occur (irreversibly) in time. At any stage (i.e. time interval) during the life history, one or more life cycle phases' activities of one or more change processes may occur.

Many architecture frameworks display life history aspects implicitly, i.e. while not having a dedicated concept, they still acknowledge a succession of steps during the life cycle of the modelled entity. The past succession of steps is irreversible. Based on the entity's history however, one may (and should) make appropriate selections from the pool of life cycle phases and thus influence the *future* life history of the entity. Of course, the future life of an entity is only partially determined; therefore, it should be feasible to represent and analyse *possible future life histories*, as may be represented on alternative life history diagrams.

Conversely, the addition of the life cycle phase dimension to the timeline representations that already exist in some architecture frameworks (e.g. as shown for C4ISR in Fig. 3.16) would give a more detailed picture of an enterprise entity's life history. This improved understanding can be achieved by showing the cause-effect relationships of life-cycle activities (as represented in Fig.

3.10), thus developing a better understanding of the past and enabling more accurate forecasts regarding its future.

3.4 The Modelling Frameworks of Reference Architectures

The GERA Reference Architecture includes the definition of the GERA *Modelling Framework*. The significance of the modelling framework resides in the fact that it defines the *scope* of potential models that change processes may need to use, or to produce. Therefore, the explicit (and implicit) scope of modelling frameworks will be investigated within each of the reviewed Architecture Frameworks. Since the granularity of individual frameworks in this respect is not identical, there are two questions to be asked:

- What explicit guidance is provided by each particular framework about the potential list of models that might need to be created during the course of various change processes[78];
- To what extent is each reviewed framework able to accommodate the complete scope of modelling as defined in the GERA Modelling Framework? This represents the implicit ability of the framework to accommodate any model that might be required, even though the framework gives no specific help for the user to identify that model as a potential candidate deliverable.

The answer to the first question is addressed both in this Section and Section 3.6. The second question is dealt with in this Section by mapping the modelling framework of each reviewed architecture against GERA's modelling framework.

NB The reader should also note that irrespective of the specific way in which a given architecture's modelling framework decides to subdivide views, these subdivisions *must* be views of an *integrated metamodel*. Thus, the way views are subdivided is less important, as long as the combined scope of the modelling framework is *complete*. For example, it should be possible (using e.g. an enterprise engineering tool) to look at the collection of integrated enterprise models through any of the views defined in the investigated modelling frameworks[79].

[78] e.g. a higher granularity modelling framework gives more specific advice on the *kinds of* models compared to a low granularity modelling framework.

[79] this is presently not possible because of the lack of a commonly agreed metamodel and ontology for integrated enterprise models.

3.4.1 PERA and Modelling Frameworks

As mentioned in Section 3.2.1, the PERA diagram[80] may be viewed as the model of an enterprise engineering methodology. Broadly speaking, the life cycle architecture of an entity[81] describes artefacts and relationships among these, where the artefacts may be various level design documents. PERA displays a two-dimensional structure comprising life cycle phases (refer Section 3.2.1) and the scope of the artefacts delivered in those phases. With each of these artefacts one can associate an activity that produces that artefact, such as a design process, an implementation process, etc. PERA distinguishes two major aspects in the life cycle of an enterprise, namely the enterprise mission accomplishment (manufacturing processes, and processes to provide services to the customer) and the enterprise 'information and control' (i.e. management and control information and processes). In PERA the information content and the processes (functions) of management and control are represented together, while other modelling frameworks (e.g. those of GRAI, CIMOSA) subdivide these into separate views. This approach is mainly due to PERA's view that the 'Information Architecture' (i.e. the structure of information collected, stored and retrieved) is closely linked to the 'Control Architecture' (i.e. the structure of management and control processes), meaning that the ultimate purpose of information is for it to be used for management and control.
Starting at the preliminary design level, these aspects are subdivided in order to accommodate the fact that the role of the human in the enterprise must be able to be represented and distinguished from all other roles taken by machinery. This explicit distinction is one of the important contributions of PERA to the enterprise integration field.
For a more in-depth coverage of PERA refer (Williams , 1994a). The mapping of PERA onto the GERA cube[82] is shown in Fig. 3.18.

---

[80] also referred to as the PERA 'windchime' (Li and Williams , 1994) due to its shape.

[81] generally, the term 'architecture of an entity' could mean two things: either the artefacts and their relations (in which case artefacts might be interpreted as parts of the system) or the artefacts created during the process of the entity's creation or change, and their relations (in which case artefacts are design documents, models, etc). The first meaning applies to type 1 architectures, while the second meaning applies to type 2 (*or life cycle*)architectures. The 'life cycle' qualifier used eliminates this ambiguity.

[82] the term 'cube' used here interchangeably with 'modelling framework'.

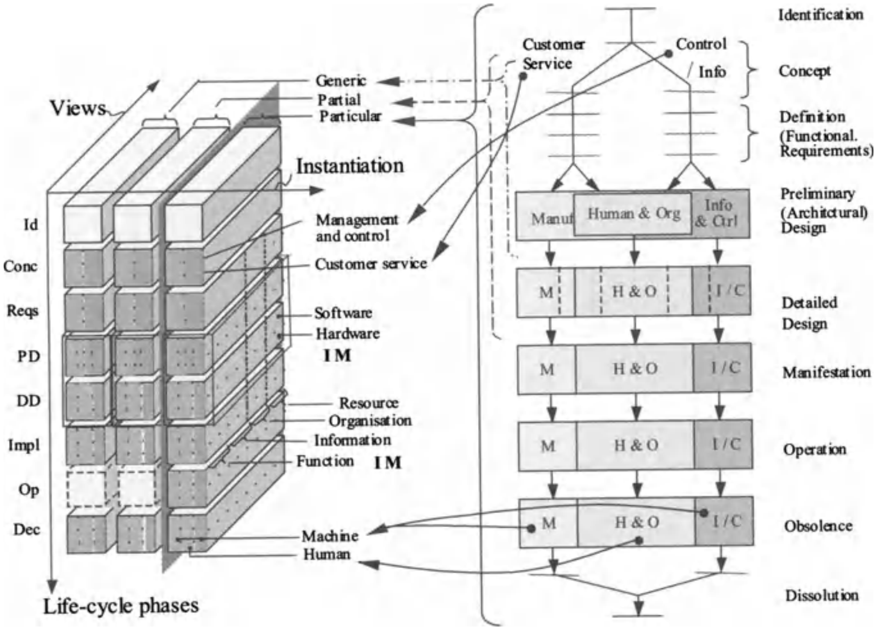


Fig. 3.18. PERA modelling framework mapping to GERA[83]

Notes about the mapping in Fig. 3.18:

- PERA advocates against the standardisation of other views (such as Functional / Information / Resource / Organisation, Software / Hardware, etc) (Li and Williams , 1994), but does implicitly cover the scope of these views. Therefore, the expression 'implicit mapping' (IM)[84] has been used in these cases.
- a genericity dimension may be identified in PERA, such as the one described in (Li and Williams , 1994) and (Williams and Li , 1998b). This enables a mapping of the PERA modelling framework onto the front face of the GERA cube[85].

NB The PERA authors consider that the Identification, Detailed Design, Manifestation and Operation life cycle phases do not have Generic or Partial components (Li and Williams , 1994). This fact has been taken into account in the mapping shown in Fig. 3.18 via the dashed and dash-dotted lines. It can be argued however, that PERA as a modelling framework applies to *all*

[83] dashed and dash-dotted brackets and lines indicate the applicability of PERA to the GERA genericity dimension, as espoused by the PERA authors.

[84] whereby the dimension/view in question is not explicitly addressed, but elements do exist in the framework

[85] since PERA explicitly diplays *two* dimensions: life cycle and genericity, which also compose GERA's 'face'.

levels of genericity as defined in GERA. For example, ontologies and partial models (the latter, potentially very specialised) about the detailed design[86] and manifestation phases (and in fact about all of the life cycle phases except for the Operation) may exist.
In reading Fig. 3.18 one must realise that once again GERA acts like a generalised framework, providing *placeholders* onto which the PERA contents may be mapped. Thus, with the exception of the 'front view' of the GERA cube[87], PERA leaves it to the user of the architecture (or to a PERA-compliant methodology) to identify the set of models that need to be produced throughout the life cycle phases. While in its scope PERA is 'complete' in the GERA sense, in its detail it relies on the methodology followed.

3.4.2 GRAI and its Modelling Framework

GRAI promotes two main principles: the importance of the decisional aspect of a manufacturing system and the necessity of a two-stage design process (user- and technically oriented) resulting a set of functional specifications. This is reflected in the structure of the GRAI-GIM modelling framework, which contains a decision view and a split structural level (the equivalent of which is the split design level in the GRAI Structured Approach shown in Fig. 3.3).
Figure 3.19 shows a possible mapping of the GRAI-GIM reference architecture onto the GERA modelling framework (life cycle phases are omitted for clarity). The GRAI modelling framework and its associated methodologies emphasize the importance of having two separate sub-phases (i.e. user-oriented and technically-oriented) in the design phase. This is due to the necessary transition from user- to system requirements, which is also reflected in the change of views (from the four: function-, information-, resource/physical- and decisional *views*, to the three: organisation-, information technology- and manufacturing technology *domains*). A possible mapping[88] of this transition is shown in Fig. 3.18.
The GRAI modelling framework, as defined in (Chen et al. , 1997), contains an Abstraction dimension (Fig. 3.19), which reflects the specialisation of the models produced as one approaches the implementation life cycle phase. However, this dimension is not reflecting genericity in the GERA sense. For example, by using it is not possible to build either a generic or a partial model of the detailed design or the implementation of a system. Still, such models are possible: e.g. a partial model of the detailed design and implementation

[86] for example, a model that describes the Detailed Design level of the ORACLE DBMS or the Apache Web Server may be considered a partial model, which will be configured (in the Manifestation phase) to create an ORACLE DBMS or Apache installation.
[87] refer (Williams and Li , 1998b) for a mapping of PERA on the 'face' of the GERA cube.
[88] note that the GRAI *views* and *domains* exist at different levels of specialisation.

would apply to a *type of* system, while the generic model of the above would contain the meaning of all of the derived models (partial and particular).

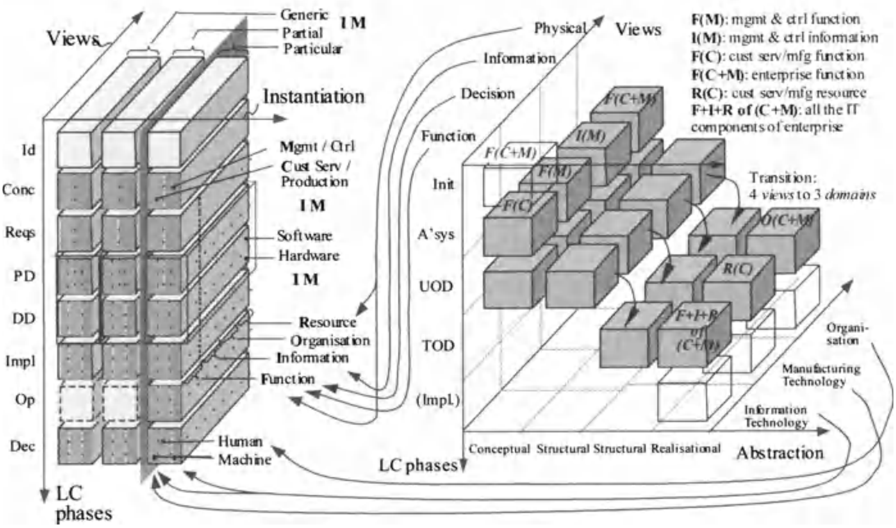


Fig. 3.19. GRAI-GIM 's modelling framework mapping to the GERA cube.

Notes about the mapping in Fig. 3.19:

- IM stands for Implicit Mapping, whereby the dimension/view in question is not explicitly addressed, but elements do exist in the framework;
- Init=Initialisation; A'sys = Analysis;UOD=User Oriented Design; TOD=Technically Oriented Design and I=Implementation;
- The Initialisation and Implementation life cycle phases have been shown since although not explicitly shown on the original GRAI framework (Chen et al. , 1997), they are addressed in the GRAI structured approach (Doumeingts , 1998a).
- Machine vs. Human, and Software vs. Hardware aspects are generally covered together[89].

The Initialization phase includes several steps (such as the overall definition of the company and the study domain definition (Chen and Doumeingts , 1996)), but does not detail additional views (such as functional, information, etc). This phase also produces a high-level *functional* description of the domain of study and since it does not detail any additonal views it may be mapped in its entirety to the Functional view of the GERA Identification and

[89] i.e. the coverage of GRAI satisfies the scope of GERA in this respect but does not implicitly define the difference. Also refer (Bernus et al. , 1996c).

Concept life cycle phases. In this phase GRAI does not separate the service and management functions (similarly to GERA, PERA and other frameworks).
On the Analysis level (matching the Requirements life cycle phase of GERA) , GRAI covers only the functional and information views - where the GRAI 'functional view' refers to the customer service and manufacturing function, the GRAI 'decisional view' refers to the management & control function and the GRAI 'information view' refers to information requirements of the service to the customer / manufacturing as well as management & control information requirements.
The three *domains* (subsystems) present at the GRAI Technical Oriented Design life cycle phase[90] are different from the matching subsystems identified by GERA views. GERA splits the system into the Organisation (human implemented subsystem), a Customer Service and Production subsystem, and a Management and Control subsystem. Thus, GRAI subdivides the automated part of the enterprise into Information Technology and non-Information Technology, while GERA subdivides the automated part of the enterprise according to the purpose – service to customer vs management and control. However, the *union* of both subdivisions is the same, thus the scopes of GERA and GRAI are the same. In Fig. 3.19 therefore, the mapping of the GRAI Information Technology System includes both management & control system (which is exclusively information processing in nature) and information processing parts of the customer service and production system.
The Physical view in GRAI refers to the physical subsystem of the enterprise. From its description in (Doumeingts , 1998a) and (Chen and Doumeingts , 1996) and the languages recommended by GRAI to model this view, it results that the Physical view maps onto the Functional, and partially Resources views of GERA.
The Decision view is an important, but generally less addressed aspect brought in by GRAI-GIM. The decisional aspect of an enterprise is modelled using a dedicated set of constructs arranged in a matrix, called the GRAI Grid (refer Fig. 3.49). The GRAI Grid represents decision centres (in fact decisional *roles*), which may be further detailed using GRAI Nets[91] or the IDEF0[92] language. This reveals the *functional* character of the decision view. However, once the decisional structure has been modelled, the GRAI Grid may be put to further use in order to (re)design the *organisational* structure of the enterprise management. This is accomplished by mapping the organisational personnel (the human *resources*) to the roles previously defined in the GRAI-Grid. The completed mapping will therefore contain information relevant to the func-

[90] GRAI Technical Oriented Design matches the GERA Preliminary (Architectural) Design life cycle phase (refer to Section 3.2.2)

[91] which are in fact the definitions of the modelling languages used to create the models in the partial and particular levels.

[92] ICAM (US Air Force's Integrated Computer Aided Manufacturing) DEFinition Function Modelling Language (ICAM , 1981)

tional, organisational and resource views of the GERA. For more information regarding the GRAI-Grid refer Section 3.7.2 and (Doumeingts et al. , 1998b). A possible extension of the GRAI modelling framework (adding the Genericity dimension) is shown in Fig. 3.20.

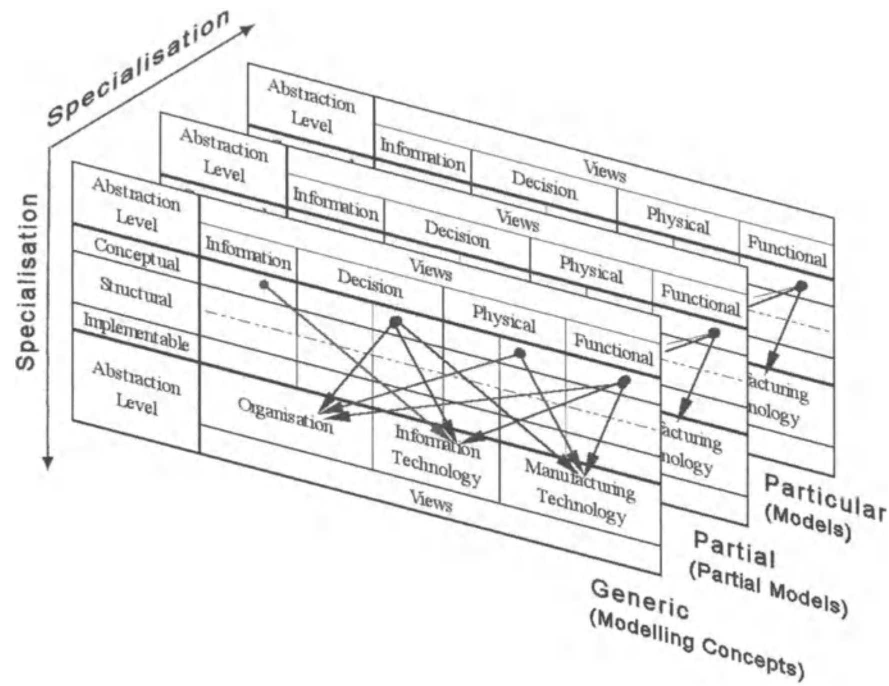


Fig. 3.20. Possible genericity dimensions of the GRAI-GIM modelling framework[93](including the 'four views to three domains' transition (Li and Williams , 1994; Doumeingts , 1998a))

Although the same framework is being used at all levels, the actual content of the framework will be different at each level of genericity. At the particular level the framework is populated with fully developed, particular models. At the partial level the framework would be populated with partial models displaying various degrees of specialisation. At the generic level, the framework would contain modelling constructs and the rules to combine them into modelling languages[94].

[93] using a flattened view (the Abstraction-Views plan in Fig. 3.19) of the GRAI modelling framework, derived from the GRAI Structured Approach.

[94] according to GERAM, these constructs and rules should ideally consistently be expressed in terms of ontologies, metamodels and glossaries.

NB Some existing GRAI-GIM models already fit in the generic and/or partial levels, such as generic production models or partial function, information, decision and stock/resource models (Bernus et al. , 1996c). However, further study is necessary in order to map these models comprehensively to GERA.
In addition, Fig. 3.20 shows the increasing degree of specialisation as one advances towards the particular models. Speialisation (seen here as a decreasing level of genericity[95]) also increases as one approaches the implementable abstraction level within the *same* level of genericity. This concept is implicitly applicable to all the modelling frameworks of the reviewed life cycle architectures.

3.4.3 The CIMOSA Modelling Framework

The straightforward mapping of the CIMOSA modelling framework to GERA reflects the origins and history of GERA[96] (refer Chapter 2 for details). One must, however, note that while the items described by the CIMOSA cube fit into the GERA placeholders, they do not necessarily fill them[97].
ENV/ISO 19439 (as the extension of the CIMOSA modelling framework) has recently been adopted by the CIMOSA association, and is based on the GERA modelling framework, thus a 1:1 mapping onto GERA is possible. The only difference is that the CIMOSA Association did not mandate the subdivision of modelling views according to human vs machine, service to customer vs management and control, and software vs. hardware. However, the 'new CIMOSA' still implicitly covers these views.
As opposed to PERA (which argues that views unnecessarily complicate the models) and Zachman (which maintains that the human perception is optimal for two-dimensional artefacts), CIMOSA uses three dimensions and introduces several views as a means to concentrate on various aspects and reduce the complexity of modelling the enterprise. CIMOSA supports an evolutionary modelling approach, where people with different type of knowledge may independently and incrementally model different aspects of the enterprise. The consistency between models and the integrity of the overall model

[95] for example: in the development of a new product, the models that fill the GRAI framework become increasingly specialised as one moves from the Conceptual to the Structural- and Implementable levels. The models become less abstract (as decisions are being taken and choices are being made, and thus more details become known) but at the same time more specialised (since they are applicable to a smaller range of products). Eventually, the models are *instantiated* in the Implementation phase (not covered in GRAI), thus producing *particular* models. An instantiated (particular) model has all variables known, except maybe for the run-time variables.

[96] CIMOSA has been one of the major contributors to the generalisation of enterprise reference architectures.

[97] in an analogy, if GERA was a bookcase with shelves for book categories, then CIMOSA would provide books for some of the shelves, without necessarily filling them up. The same analogy is valid for other reviewed architecture frameworks.

is maintained by the set of integrated CIMOSA metamodels of the CIMOSA modelling framework[98] (this framework is shown in the right-hand side of Fig. 3.21).
CIMOSA provides a reference architecture composed of generic and partial levels. The generic levels contain building blocks such as Domain Relationship, Capability Set, Objective / Constraint etc.(AMICE , 1993). These blocks contain the constructs and define the meaning of the CIMOSA languages (refer Section 3.5.3).
CIMOSA enforces neither the necessity to model all life cycle phases nor a particular sequence in modelling[99]. It also allows the extension of the set of views provided as needed[100].
Notes about Fig. 3.21:

- F, I, R, O represent the CIMOSA Functional, Informational, Resources and Organisation views;
- IM stands for Implicit Mapping, whereby the dimension/view in question is not explicitly addressed, but elements do exist in the framework;

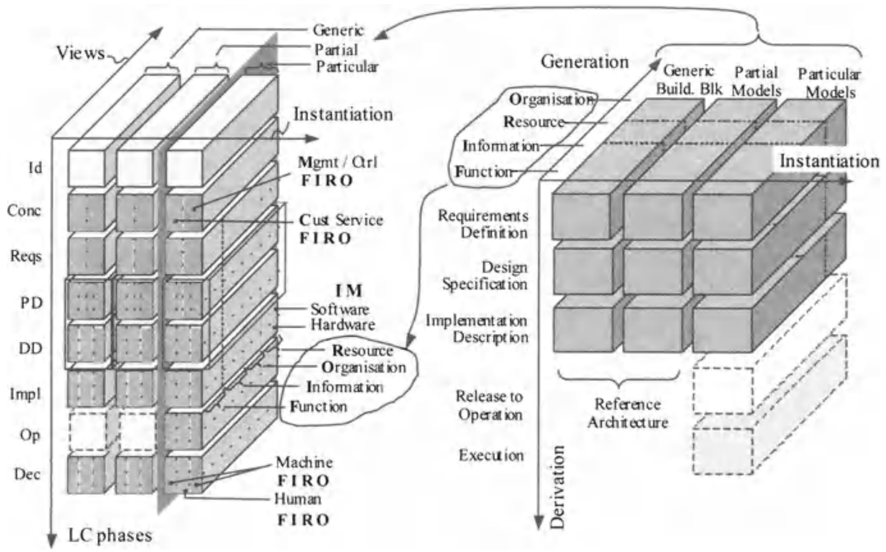


Fig. 3.21. Modelling framework mapping: CIMOSA to GERA

[98] thus, the models are views of the CIMOSA metamodels, which in turn may be described in a high-level metamodel.
[99] rightfully so - life cycle phases (as opposed to life history stages) should not include any temporality or succession aspects.
[100] for example, CIMOSA allows the extension of the Organisation- and Function views in order to include human competencies- and decisional aspects.

Examples of mapping acronyms and their intended interpretation:

- Management and control: FIRO = management and control view is represented in CIMOSA in the Functional, Informational, Resource and Organisation views[101].
- Customer Service: FIRO = the Customer Service view of GERA is represented in the Functional, Informational, Resource and Organisation views of CIMOSA[102].
- Human and Machine: FIRO = the Human and Machine aspects of the enterprise are taken into account in CIMOSA in its Functional, Informational, Resource and Organisation views[103].

The objective of the CIMOSA architecture framework is to organise enterprise models that are needed for model driven (event-driven) control of enterprise processes. Thus, the CIMOSA modelling framework in its *intended* scope prepares to accommodate models necessary to achieve this. Therefore, while the mapping onto GERA appears to be 1:1, in fact a fully populated CIMOSA model would not cover the scope of modelling of GERA, simply because certain detail does not need to be modelled in order to achieve model-based control.

The *potential* scope of the CIMOSA modelling framework is the same as that of GERA (even though CIMOSA makes only implicit differentiations between some aspects). The *actual* scope of CIMOSA is determined by modelling potential (expressive power) of the enterprise modelling languages proposed by CIMOSA, which is significantly less than the potential scope, due to the standardisation of a set of modelling constructs by the CIMOSA Association. Naturally, the advantage of having a standard set of modelling constructs is that vendors can produce standard modelling tools products for the support of CIMOSA users.

Typically however, some tasks executed by humans are ill structured, nondeterministic and cannot be completely formalised. Task descriptions for humans vary widely according to the implicit knowledge of the target audience[104]. CIMOSA regards humans as resources with capabilities (lately including competencies (Berio and Vernadat , 1999)) and responsibilities. However,

[101] the CIMOSA modelling language constructs do not explicitly cover management and control at Implementation Description, Generic Building blocks level (Bernus et al. , 1996c). NB management and control in this case mainly covers enterprise software only.

[102] the CIMOSA modelling language constructs only cover the Requirements Design level and only for the Partial and Particular Models (Bernus et al. , 1996c).

[103] the CIMOSA modelling language constructs only cover the Human aspect at the Requirements Design level (where the machine aspect is not covered)

[104] the level of detail of the task descriptions ('human software' in GERA terms) would normally be inversely proportional with the skills (or competency) of the human target.

aspects such as the place of the human within the enterprise or the decisional structure[105] of an organisation are not clearly modelled (even though the modelling framework *would* be able to be populated with languages that allow this). Therefore, enterprise modellers seeking to model *all* aspects of the target enterprise in the sense of its *potential* scope need to refer to additional modelling frameworks, which complement the above-mentioned views (such as human / decisional, software / hardware, customer service / management and control)[106].

3.4.4 Zachman's Modelling Framework

The Zachman Framework initially addressed the information subsystem of an enterprise[107] (Zachman , 1987). It also appeared to be restricted only to the modelling framework component of an architecture framework. Subsequently, the scope of the framework has been extended to cover aspects informally defined in the previous version (Sowa and Zachman , 1992). This extension and formalisation, together with the emergence of modelling methodologies and partial models have provided for the evolution of the original Zachman *modelling* framework into the Zachman *architecture* framework[108].
A mapping of the Zachman framework 'abstractions' onto the GERA Function / Information / Resource / Organisation views is shown in Fig. 3.22. Generally speaking, architecture frameworks acknowledge informational aspects, since there is a good general understanding of the data modelling paradigms and processes. Zachman is no exception - therefore, the GERA Information view maps onto the Data abstraction in Zachman.
A closer look at the Zachman framework reveals however that the remaining GERA views are not mappable 'one-to-one' to the Zachman framework. Brief explanations of this aspect are provided below.

- the People abstraction in Zachman is mappable to the GERA Organisation view, since they comprise a similar scope. However, people are also a (human) resource - therefore the People abstraction maps onto the Resource view in GERA as well[109];

[105] the decisional view is in fact a specialised functional view (using specific constructs).

[106] such as e.g. PERA for human / non-human aspects, or GRAI for decisional aspects

[107] the information subsystem (or information system) has always been part of enterprises (and has been enabled by the information technologies historically available) but has recently dramatically increased in importance.

[108] the initial name of the Zachman framework, i.e. 'Framework for Information Systems Architecture' (Zachman , 1987), was subsequently changed to 'Framework for Enterprise Architecture' (Zachman , 2000a), possibly in order to reflect and support this evolution.

[109] note that the GERA modelling framework (inherited from CIMOSA) considers the Organisation as the relation between human resources (people) and functions

- the Zachman Function abstraction covers the domain type defined by the GERA Function view. However, the Zachman Motivation abstraction provides for rule modelling, which may also be mapped onto the Functional view in GERA[110];
- The Network abstraction in Zachman covers the non-human resources - therefore, together with the People abstraction, it maps onto the GERA Resource view.

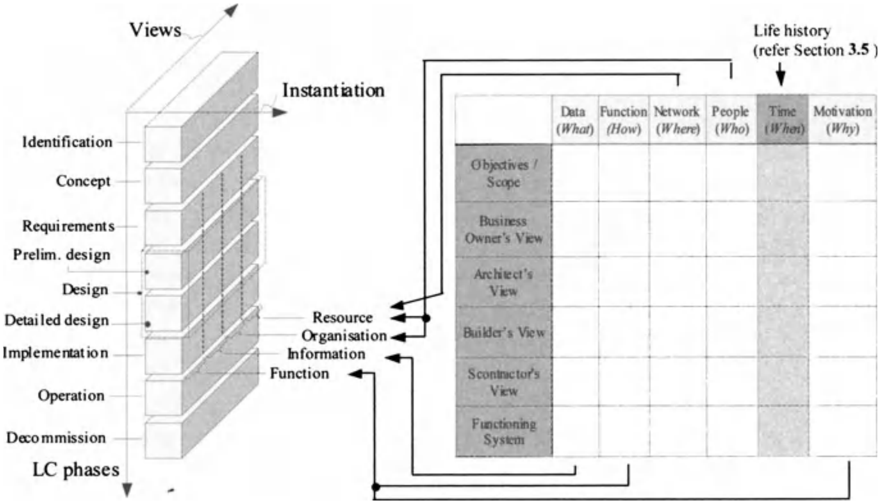


Fig. 3.22. Modelling framework mapping using the GERA F, I, R, O views

The mapping of the Motivation abstraction on the GERA Function view deserves an extended explanation. Rule modelling is part of behaviour modelling, which in its turn is part of extended functional modelling. When modelling the functional aspects the following degrees of detail may be observed (not necessarily in the order shown):

- activities are shown with their inputs and outputs (e.g. using IDEF0) which are objects, but no objects' details are shown;
- events are added (e.g. using Petri Nets, Harel State Charts, State Transition Diagrams, etc) so as to model system behaviour / dynamics (reaction to events)

(roles that humans take). Separately from this, Resources (including human resources) are modelled in the Resource view (i.e. human resources are identified and their capabilities are described).

110 modelling languages used for Function modelling usually allow (or require) an up-front statement – as part of the model – of the modelling context, describing why and for what purpose the model was constructed;

- the inputs / output objects are detailed (e.g. using coloured Petri Nets[111], IDEF3 Object State Transition Networks (OSTNs), etc)
- pre- and post conditions (rules) are added (e.g. using structured English, decision tables/trees, mathematical logic)
- temporal dimension is added (timed Petri Nets[112], IDEF3 process flow description, etc)

It has therefore been considered that the Motivation abstraction represents an extension to the GERA Functional view.

Considering a mapping of the Zachman framework against the other GERA views, one will realise that even within the same column, each cell in the Zachman framework covers GERA aspects to a different extent, depending on the perspective (or row) containing them.

Take for example the Network abstraction. At the Objectives-, Business Owner's view- and Architect's view levels, the Network abstraction makes no difference between the mission accomplishment and mission control aspects of the enterprise. At the Architect's- , Builder's-, Subcontractor's- and Functioning System views, the Network abstraction coverage is limited to non-human aspects of the resources' locations or structure, however at the same time covering both hardware and software aspects.

Another example is the People (Who) abstraction. The cells in the upper (Objectives and Business Owner) perspectives mainly relate to the human aspects, therefore using concepts such as organisational units, mission, organisational chart, etc. However, moving down to the Architect's, Builder's and Subcontractor's perspectives, the focus gradually shifts towards the human - machine interaction and subsequently to interface specifications (away from the human aspect).

The problem with this approach is that the user cannot readily see (a) all the possible aspects that *could be* covered and (b) which of those possible aspects *are* indeed covered at each perspective. A partial solution is to explicitly specify this information in each cell by attaching separate descriptions to the cells in question[113].

One other solution is to use the GERA multi-view approach and create subdivisions inside each cell of the Zachman framework corresponding to each

---

[111] a specialisation of Petri nets where tokens own attributes called *colours*. Refer Jensen (1992) for details.

[112] a specialisation of Petri nets including temporal aspects (i.e. a duration is associated to each transition or each place).

[113] this is a solution often used when the modelling language used does not possess the necessary expressiveness for the particular modelling task. Example: the ER and UML modelling languages cannot fully express constraints - therefore they are explicitly specified via text or stereotypes. The more formal solution is a separate, dedicated language - e.g. the Object Constraint Language (OCL) in the case of UML.

of the applicable GERA views.. A filled cell subdivision would then indicate that the matching GERA aspect is covered within that cell.

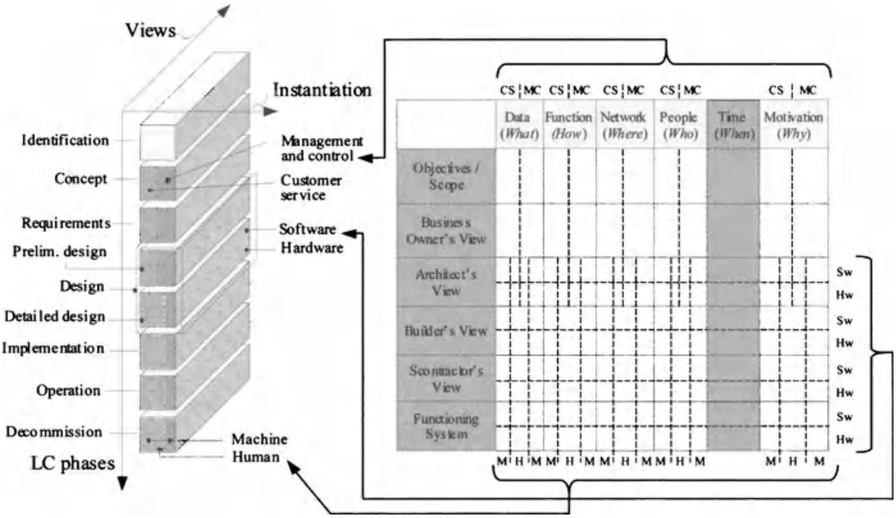


Fig. 3.23. Possible additional views in Zachman according to GERA

Notes about Fig. 3.23:

- M, H denote the Machine and Human aspects in GERA;
- Hw, Sw are the GERA Hardware and Software aspects;
- CS, MC are the Customer Service and Management / Control views in GERA.

Figure 3.24 presents an extract (one single cell) of the right-hand side of Fig. 3.23, covering the Architect's view of the Network abstraction and using the descriptions provided in (Zachman , 1987).
The represented cell explicitly covers the Software and Hardware aspects of the Non-Human (Machine) side of the Management and Control of the enterprise[114]. This extract is presented primarily to illustrate the potential use of the GERA views to further scope the Zachman framework cells and to identify potential coverage gaps and overlaps.
NB Although the Zachman framework considers the Network abstraction to cover location and structure, in this example the Hardware description of the Human side of both Management / Control and Customer Service in Fig. 3.24 has been limited to Location, mainly due to the presence of the People

[114] the software and hardware non-human side of the Customer Service (manufacturing) may also be considered as covered, although this is not explicitly stated in the reference used.

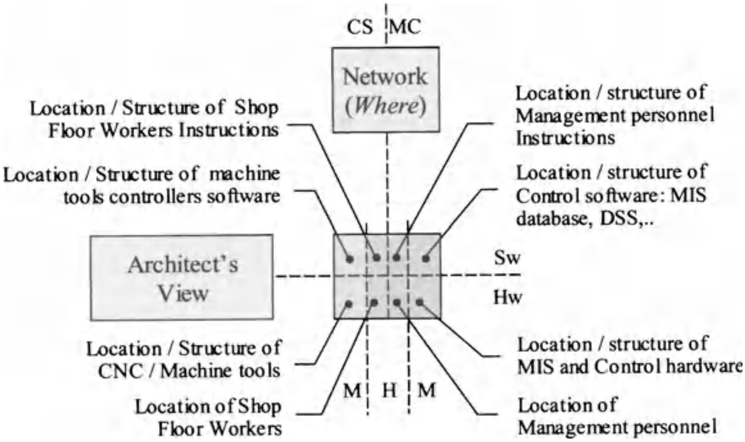


Fig. 3.24. Extract from Fig. 3.23

abstraction, which may be more appropriately used to represent the *structure* of the enterprise personnel (organisation).

The Zachman framework does not explicitly contain a genericity dimension. However, separate frameworks may be used for the three main areas of genericity (generic - partial - particular), similar to GERA and to GRAI (refer Section 3.4.2 and Fig. 3.20). In doing so however, the 'Functioning System' perspective should be omitted from the generic and partial frameworks (the shaded areas in Fig. 3.25). Similar to the case of GRAI, the particular framework would contain fully developed models, while the partial Zachman framework would contain reference models[115]. The generic level framework would then contain metamodels - modelling constructs and their relationships, ontologies and glossaries.

Referring to Fig. 3.25 and similar to Fig. 3.20, there is another (implicit) genericity dimension *along* the Perspective axis. The views of the people involved in the life cycle phases of the artefact may be "of a different nature" (Zachman , 1987) but they do refer to the same artefact (since they belong to the same framework) on a different level of detail. For example, a high level user requirement in the owner's view may translate into several system requirements in the architect's view, become implementable by using a combination of technologies / materials / tools described in the builder's view and physically be fulfilled by putting together the components built in the subcontractor's view.

Each representation may use specific means of expression, but in essence they refer to the same artefact in various degrees of detail. The degree of speciali-

[115] these may be models of prototypes, 'class models' or models of the common parts of a set of enterprises, according to the GERA concept of partial model.

sation of the views (models) will gradually increase as one moves closer to the functioning system view (downwards, in the Zachman framework). This happens because decisions have to be made regarding various system parameters (e.g. the kind of architecture to be used, processes / technology / human skills to be employed etc). Instantiation then occurs when abstract representations become reality. This only occurs in the Builder / Subcontractor View of the *particular level* Zachman framework (mapped onto the GERA Implementation phase (refer Fig. 3.5).

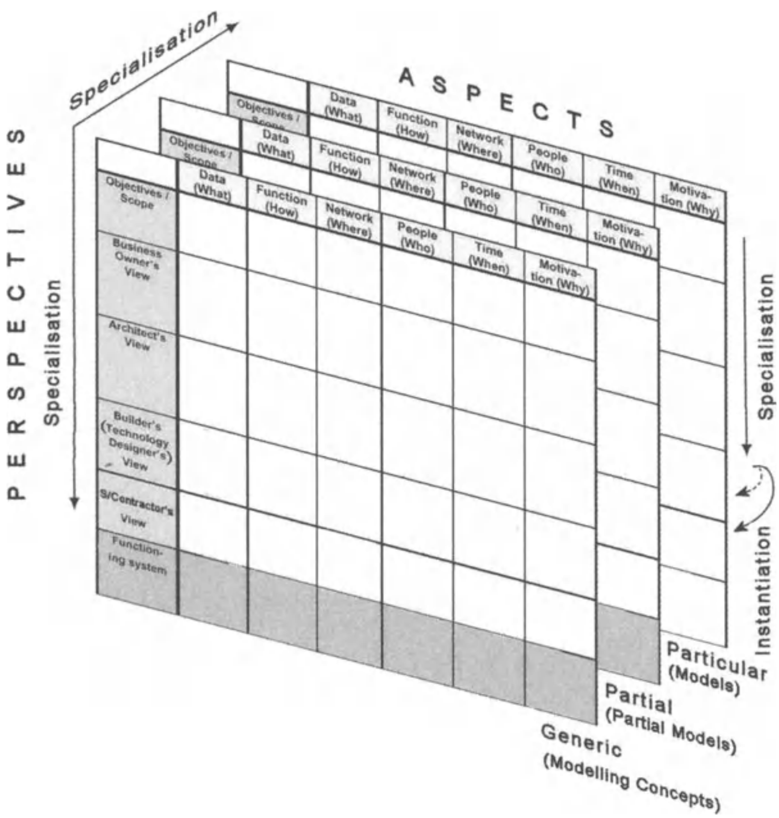


Fig. 3.25. Possible genericity dimensions in the Zachman framework[116]

The Zachman framework also appears to include a third, implicit genericity dimension derived from its recursiveness, as subsequently shown in Section 3.5.4.

[116] shaded areas mark the non-applicable views; the instantiation shown only applies to the particular Zachman framework.

3.4.5 C4ISR's Modelling Framework[117]

The evaluation of the C4ISR's modelling framework from the GERA perspective is not a trivial task[118].In understanding the C4ISR components, one must take into consideration the origins of- and the influences on the architecture in question. Levis and Wagenhals (2000) for example argue that C4ISR displays a strong Structured Analysis (SA)[119] character, which would explain some of its architectural products and views[120]. Object Oriented influences are also present in C4ISR's architectural products, partial models and development trends.

A mapping on GERA is possible taking into account the features and descriptions of the C4ISR artefacts (architectural products, resources, etc). C4ISR divides its architectural products into *essential* and *supporting.* The essential products must always be present so that the architecture may be effectively communicated to the intended audience. C4ISR maintains that not all supporting views are always needed[121], depending on the envisaged purpose of the architecture.

The use of the GERA modelling framework as a reference may be of great help in (a) defining which C4ISR products are needed for a specific modelling task and (b) identifying aspects possibly still uncovered after the selection of the C4ISR products for that task. Any gaps identified in step (b) may then be filled by selecting products from other architectures, using the same reference (GERA) to maintain the consistency of the selection process.

The following figures show the use of GERA to identify aspects covered by the C4ISR products. **NB** although graphically all mappings are shown pointing towards the management and control side of the GERA modelling framework, C4ISR does not explicitly differentiate[122] between management and control vs. customer service, or human- vs. machine aspects. Where applicable, the actual coverage of the C4ISR products has been shown (e.g. in Fig. 3.27). A brief justification of the C4ISR mapping onto the GERA (shown in Fig. 3.26) follows.

[117] the contents of this Section has also built on material contained in (Bernus , 2000)

[118] C4ISR's approach is directly supported by very few tools, and no current Systems Engineering formalisms.

[119] refer (DeMarco , 1979) for details on the Structured Analysis approach.

[120] such as the Operational concept and view, Systems view (equivalent to the Physical Architecture in SA), etc.

[121] this C4ISR recommendation is disputed by Levis and Wagenhals (2001b). They argue that all products are highly recommended and that the C4ISR use of the term 'supporting' gives the wrong impression of optionality.

[122] in the reviewed C4ISR documents. A comprehensive review of *all* of the current C4ISR associated documents (which is beyond the scope of this Chapter) may reveal additional aspects and should be the focus of a subsequent study.

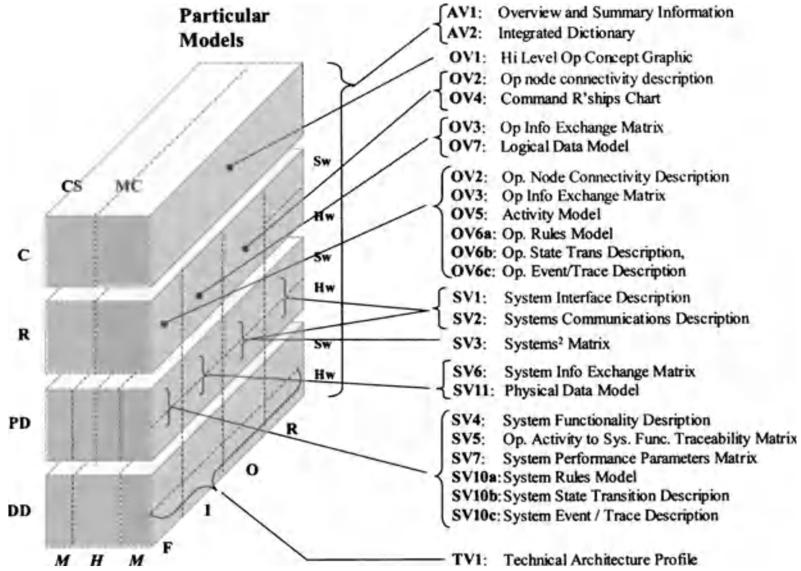


Fig. 3.26. C4ISR modelling framework mapping to GERA Particular Models level[123].

The Overview and Summary Information (AV-1) refers to all views - therefore it maps onto all the GERA life cycle phases. The Integrated Dictionary (AV-2) is in fact a glossary, which is in direct relationship to the architecture metamodel. In this respect, AV-2 is a Generic Enterprise Modelling Concept (GEMC), which is described in Section 3.8.3.5. AV-2 applies to all views of the architecture and may hence be mapped onto all applicable GERA life cycle phases.

The High Level Operational Concept Graphic (OV-1) is defined by C4ISR as a 'facilitator of human communication'. It represents the possible answers to a perceived need for an artefact or its change. As such, it maps onto the Concept life cycle of GERA. The Operational Node Connectivity Description (OV-2) describes operational nodes / elements and requirements for their connectivity. Therefore, OV-2 describes requirements for the organisation of the artefact being architected and maps onto the Organisation view of the GERA Requirements lifecycle phase[124].

123 mappings cover both Customer Service and Management / Control aspects, although this was omitted in the figure for clarity purposes.

124 in addition, OV-2 describes functionality and hence it also maps onto the GERA Functional view, as subsequently shown.

The Command Relationships Chart (OV-4) describes high-level requirements for the relations between organisations or resources. As a result, it maps onto the Organisation view in GERA.
NB OV-4 is the only view to provide insight into organisational / decisional aspects in C4ISR, which appears to be a current shortcoming. More coverage of these aspects may be provided by adopting other modelling frameworks' approaches and/or reference models[125].
The Operational Information Exchange Matrix (OV-3) and the Logical Data Model (OV-7) both describe information. While OV-3 describes information exchange and its required attributes, OV-7 focuses on the data model of the artefact being modelled (e.g. using Entity Relationship diagrams). Hence, both OV-3 and OV-7 map onto the Information view of the GERA Requirements life cycle phase. **NB** Due to its content, it is possible that for specific modelling tasks OV-7 will also extend into the Information view of the GERA Preliminary Design phase.
The requirements for the functionality of the modelled artefact are covered in C4ISR by several products. The Operational Node Connectivity description (OV-2) and the Information Exchange Matrix (OV-3) partly contain data flow descriptions and hence describe some functional aspects. The Activity Model (OV-5), Operational Rules Model (OV-6a), Operational State Transition Description (OV-6b) and the Operational Event / Trace Description (OV-6c) refer to functional modelling in various degrees of detail. Static functional modelling is covered by OV-5, while behavioural (dynamics) modelling is covered by the rest, involving rule modelling (OV-6a), state transitions using the rules (OV-6b) and events (OV-6c).
The System Interface description (SV-1) has a dual purpose: define the *inter*-system interfaces but also the *intra*-system structure, according to the C4ISR Architectures Working Group (1997). Therefore, SV-1 maps on both the GERA Resource and Organisation views of the GERA Preliminary Design[126]. The Systems Communication Description (SV-2) attempts to identify the communication pathways / networks at both external and internal levels and therefore also maps on the Resource and Organisation views of the GERA Preliminary Design phase.
The System to System Matrix (or System^2 Matrix, code SV-3) product describes the inter-relations of the systems composing the modelled artefact - in fact, part of the artefact organisation. Therefore SV-3 maps onto the Organisation view of GERA's Preliminary Design life cycle phase.
The System Information Exchange Matrix (SV-6) and the Physical Data Model (SV-11) are the System view products matching the Operational Information Exchange Matrix (OV-3) and the Logical Data Model (OV-7) belong-

---

125 such as, e.g. GRAI Grid for decisional aspects and PERA for human aspects.

126 SV-1 describes structure (i.e. Organisation minus [task allocation]). In addition, the items described are also resources - hence the implicit mapping to GERA Resource.

ing to Operational view[127]. Therefore, they map onto the GERA Information view of the Preliminary Design life cycle phase. An observation similar to the Logical Data Model (OV-7) applies for the Physical Data Model (SV-11) in the sense that SV-11 may extend into the Information view of the GERA Detailed Design phase, depending on the specific modelling task.

The Systems Functionality Description (SV-4) focuses on the flow of data among system functions and hence it may be mapped onto the GERA Functional view of the Preliminary Design life cycle phase.

An essential aspect of modelling is the consistency between models developed in various life cycle phases. There are several (and complementary) ways of enforcing this consistency, such as the use of a modelling framework based on a metamodel / ontologies, the traceability of requirements, etc. The Operational Activity to System Function Traceability Matrix (SV-5) addresses consistency checking by providing a way to check the way the user requirements elicited in the Operational view have filtered into the system requirements expressed in the System view.

The System Performance Parameters Matrix (SV-7) provides means to specify indicators for system performance requirements for both present and future states of the modelled artefact. Performance indicators may be derived from standards with which the modelled artefact needs to comply. Therefore, it is assumed that SV-7 is influenced by the Standards Technology Forecast (TV-2). Performance expresses the dynamic aspect of functionality - therefore SV-7 maps onto the Functional view of GERA at the Preliminary Design life cycle phase.

The behavioural aspect of the modelled artefact (or system) is modelled via the System Rules Model (SV-10a), System State Transition Description (SV-10b) and the System Event / Trace Description (SV-10c). They are the equivalents of the matching products in the Operational view (OV-6a, OV-6b and OV-6c respectively) and therefore map onto the Functional view of the GERA Preliminary Design life cycle phase.

As previously shown, C4ISR defines the role of the Technical Architecture Profile (TV-1) as "the set of rules that governs system implementation and operation" (C4ISR Architectures Working Group , 1997). As such, TV-1 belongs to the Detailed Design life cycle phase of GERA and should encompass all GERA views (Function / Information / Organisation / Resource, Management / Control vs. Customer Service, Software vs. Hardware and Human vs. Machine).

The remaining C4ISR view components (SV-8, SV-9 and TV-2) relate to reference models rather than to particular models (as subsequently explained). Therefore, although they are not being explicitly described as partial (reference) models in the C4ISR framework, they can be mapped onto the partial

[127] e.g. if OV-7 were expressed as an ER model, SV-11 could be a Relational Model. As a general observation, the meanings of 'logical' and 'physical' in data modelling vary greatly from one modelling framework/method to another.

model level of GERA. The partial models[128] *explicitly* defined as such in C4ISR are covered in Section 3.7.5.
C4ISR acknowledges the fact that the standards (which are third-party- or proprietary partial models displaying various degrees of specialisation[129]) used in the modelling process are subject to change in time and that new standards impacting on the existing and future architectures will emerge. The Standards Technology Forecast (TV-2) attempts to identify emerging standards that may affect the given architecture in the near / far future[130]. As such, TV-2 maps onto the Partial Model level, Detailed Design phase of GERA and should encompass *all* aspects of the entity modelled (refer the Detailed Design phase of Fig. 3.27).

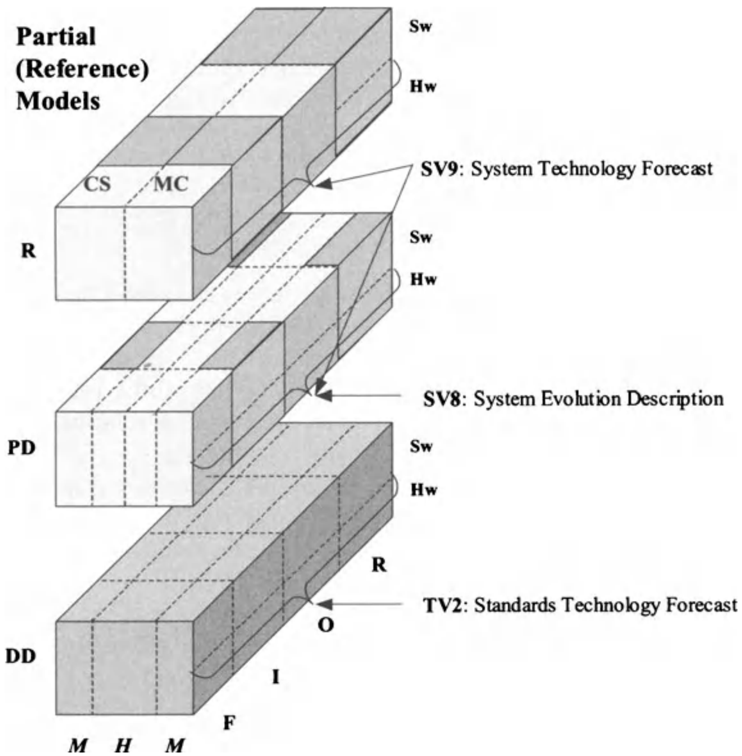


Fig. 3.27. C4ISR Standards Technology and System Forecast modelling to GERA Partial Models level [131].

[128] Universal Reference Resources, in C4ISR terms.
[129] e.g. the partial model 'ISO12207: Software Systems Life Cycle' is more specialised than (and in fact, is a specialisation of) the partial model 'ISO15288: Systems Life Cycle'. Refer (Noran , 2000a) for justification and mapping.
[130] temporal aspects of TV-2 have been noted in Section 3.3.6

The reviewed version (C4ISR v2.0) of the architecture framework includes two products that attempt to project the modelled system or entity in the future in order to identify any influences that future products/developments may have on the modelled system / entity. This is achieved using partial models[132].
The shaded areas in Fig. 3.27 covered by the System Evolution Description (SV-8) and System Technology Forecast (SV-9) reflect the fact that the examples for SV-8 and SV-9 given in the reviewed C4ISR document (C4ISR Architectures Working Group , 1997) explicitly address the Information and Resources on the Customer Service and Management / Control sides and generally the non-human aspects of the architecture. It has been assumed however that in fact predictions can (and should) be made about *all* aspects of the modelled artefact (hence the mappings in Fig. 3.27). In this respect, the GERA structure applied to the C4ISR architecture products provides a checklist against which missing architecture products may be identified (and also completed, depending on the purpose- and the target users of the architecture). Both the System Technology Forecast and the System Evolution Description have temporal connotations, which may potentially be used to model the entity's life history, as shown in Section 3.3.6.

3.4.6 Modelling Framework of ARIS

ARIS takes a similar (i.e. event-driven, process oriented) approach to CIMOSA, however having its origins in the business-oriented aspects of the enterprise, rather than in CIM. Consistent with the approach of other architecture frameworks, ARIS employs views to reduce complexity in describing the modelled entity. ARIS however, emphasizes the need to preserve the relationships between the architectural views in an *explicit* way (rather than e.g. solely relying on the underlying metamodel to integrate them). This is achieved in ARIS by means of an additional Process / Control view (refer Fig. 3.28 and Fig. 3.29). A brief justification of the mapping of the ARIS views onto the GERA views follows.
The ARIS Function view contains (as expected) functions transforming the inputs into outputs, but also goals that drive- and are supported by functions[133]. Hence, the ARIS Function view maps onto the GERA Function view.
The Organisation view of ARIS groups entities such as human output, financial resources, computer hardware, etc on the criterion of acting on a common work object. Therefore, the ARIS Organisation view maps onto both GERA Organisation and Resources views.

---

[131] mappings cover both the Management and Control (MC) and Customer Service (CS) views of GERA.

[132] since the future state of the modelled entity / system and the future existence of the factors influencing it cannot be fully defined.

[133] the goals may be linked to the Zachman framework's 'Why' column, i.e. business rules. In Section 3.4.4, business rules in the Zachman framework have been mapped onto the GERA Function view.

The ARIS Output view[134] describes software and hardware function inputs and outputs, while the ARIS Data view describes information services objects and messages that trigger-, or are triggered by functions. Therefore, from the GERA viewpoint, the ARIS Data and Output views are complementary and together they map onto the GERA Information view.

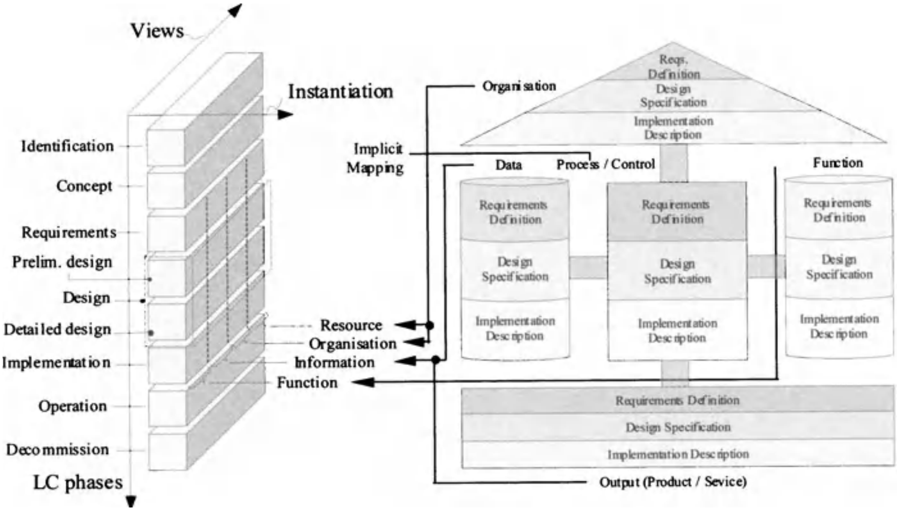


Fig. 3.28. The ARIS framework (House) and its mapping onto the GERA Particular level

The contents of the ARIS Control view must be analysed prior to its mapping onto the GERA framework. As can be seen from Fig. 3.29, the Control view does not contain any unique constructs. Its models use constructs belonging to other views, therefore creating a set of relations among the other views.

[134] the Output view has been subsequently added to the ARIS House (framework) (Scheer , 1999).

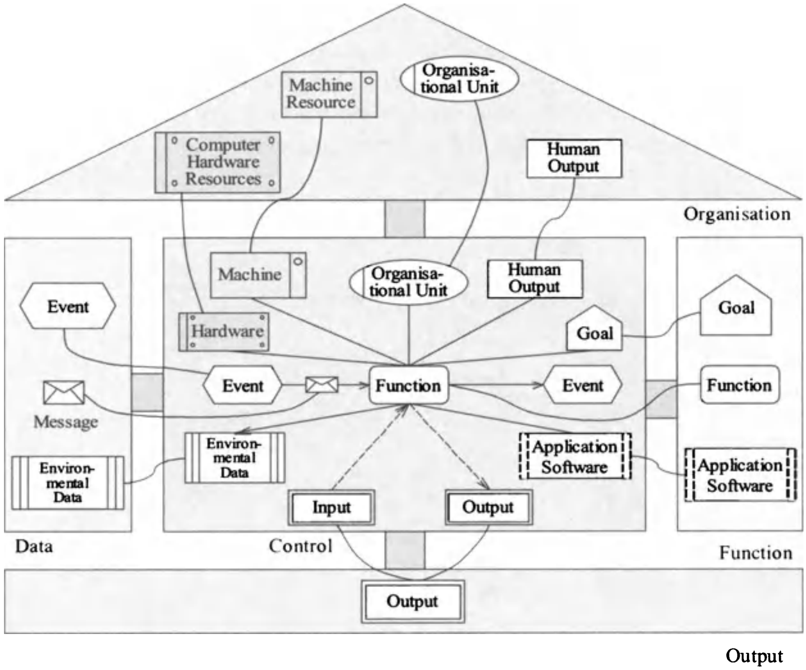


Fig. 3.29. The relation between views in ARIS[135] (based on Scheer (1999))

According to Fig. 3.29, the Control view displays the Function-Data relationship by using a function-data model, which overlaps partly with the functional model contained in the ARIS Function view and partly with the data model contained in the ARIS Data view. The same applies for all other views. Therefore, the Control view implicitly maps onto all of the GERA views. This specific structure of the Control view is confirmed by the ARIS high-level metamodel subsequently shown in Fig. 3.62 and is taken into consideration by the ARIS modelling methodology (refer Section 3.6.6).
As can be seen from the figures above, ARIS does not define a Resource view. Still, ARIS describes hardware / software and human / non-human resources in its Organisation view (which explains why the ARIS Control view also maps onto the GERA Resource view).
The ARIS framework defines modelling *levels*, which may be considered equivalents of the GERA instantiation levels (refer Fig. 3.30). The ARIS Instance level contains instances of the model constructs occurring during model ex-

[135] As a general observation, the ARIS House view contents shown in Fig. 3.29 appears to also apply to the Requirements Definition level, in which case it may be argued that some design decisions (such as e.g. regarding human / non-human tasks) are perhaps made too early (this could be rather part of a methodological debate).

ecution and as such it is not represented in the GERA framework[136]. The ARIS Application (or Model) level contains the main deliverables of the modelling effort, i.e. the models (reflecting the real world) built using the ARIS modelling constructs. This level maps onto the GERA Particular- and Partial Models[137] instantiation levels. The ARIS Meta level contains metamodels, i.e. definitions / meanings of the modelling constructs (meta-classes) and the rules of how to combine them[138].

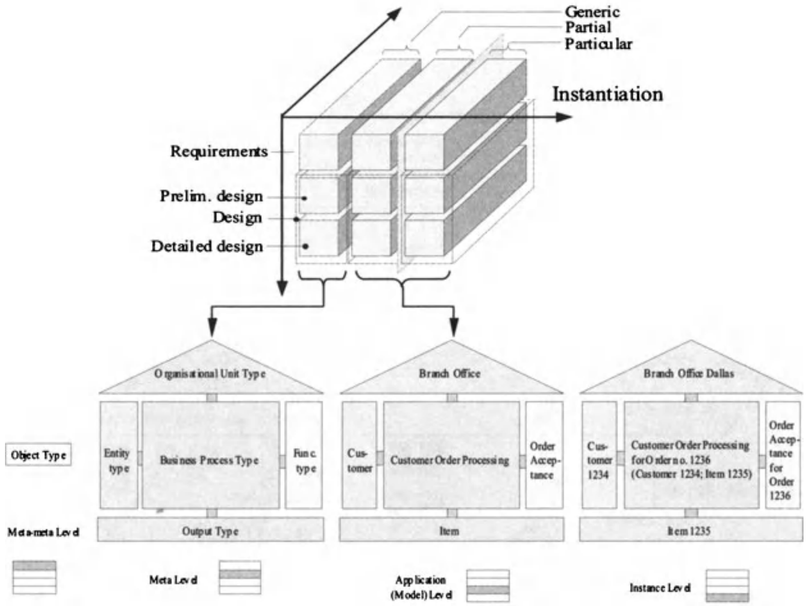


Fig. 3.30. (Genericity) levels in ARIS and their mapping on the GERA framework.

Some examples of the Meta level contents are ARIS metamodels, the ARIS structure shown in Fig. 3.29 and the database schema of the repository containing ARIS models. Consequently, the Meta level maps onto the GERA Generic level. The ARIS Meta-meta level contains only one construct, the

[136] the instance level is less relevant to the purpose of model building but more applicable towards the run-time management / operation of the modelled entity.

[137] the reader is reminded that Partial Models are incomplete models that must be either further specialised into more specific Partial Models and/or (ultimately) instantiated to become Particular Models. Therefore, models and partial models both belong to the ARIS model level.

[138] which is in essence the syntax of the modelling language.

Object Type, out of which all of the other meta classes may be instantiated[139] and which has no equivalent in the GERA framework[140].
The type of modelling level structure used in ARIS (comprising the four main levels ranging from meta-metamodel to instance) is commonly used in order to describe modelling frameworks and modelling languages[141].
In addition to the ARIS House, the ARIS architecture framework proposes a 'concept' (in fact, a possible specialisation of the ARIS House for the purpose of business process management) called the House of Business Engineering (or HOBE, shown in Fig. 3.31).

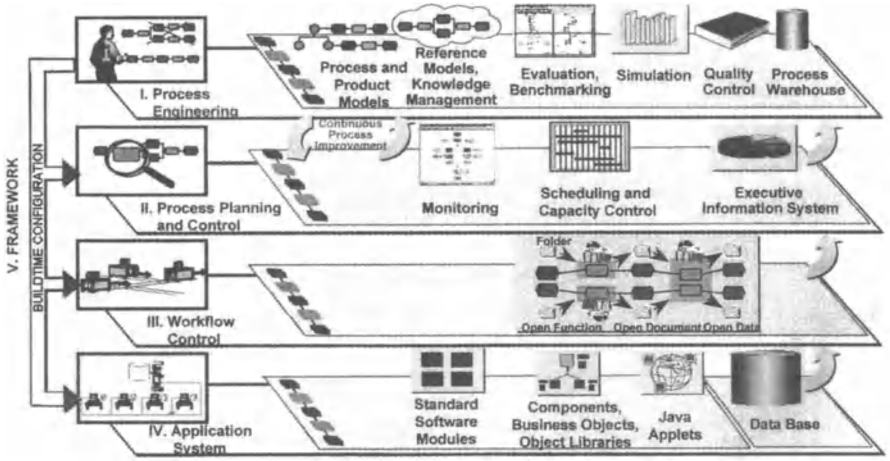


Fig. 3.31. House Of Business Engineering: An extended ARIS framework (Scheer , 1998b).

The arrows present in the HOBE framework hint towards a *succession* of tasks in modelling and managing the business processes - which in turn suggest that the HOBE also has a role in the ARIS modelling methodology. The HOBE is structured at five levels, with the fifth level representing the framework itself. The first level (Process engineering) is the most important since it provides the contents (processes) for the subsequent four levels. The second HOBE level (Process Planning and Control) focuses on the operation of the existing processes. HOBE level three (Workflow Control) aims to convert business

139 object (modelling construct) types in ARIS are stored as instances of the Object Type construct.
140 this is mainly because it is a language used to express generic models (and as such it would map onto one of GERAM's enterprise modelling languages)
141 for example the Unified Modelling Language (UML) or the Java Programming language (which is also a modelling language).

processes into IT tools and control the flow of objects through the enterprise. Level four of HOBE contains application systems, which are started (called upon) by the workflow systems in the upper level.
A mapping of the ARIS HOBE on the GERA(M) is a non-trivial task. Due to its intended use (process design *and* operation / management), HOBE contains references to various GERAM elements, such as reference models, enterprise modules, and refers to several GERA life cycle phases such as Architectural and Detailed Design, Implementation and Operation. (Scheer , 1998b) states that the life cycle dimensions of the ARIS House are perpendicular to the ARIS HOBE levels. Thus, the two frameworks may be combined into a three-dimensional structure, as further detailed in Section 3.6.6 and Fig. 3.45.

3.4.7 Conclusion About Modelling Frameworks

A modelling framework is a structure containing placeholders for artefacts needed in the modelling process. Depending on the structure of the framework, the type of these artefacts may be limited to models, or it may extend to other construct types such as partial models, metamodels, glossaries, etc. For example, the GERA modelling framework contains placeholders for artefacts whose type is described in the GERAM metamodel (Fig. 3.1).
The reviewed modelling frameworks reflect the purpose for which they were primarily and originally built: Computer Integrated Manufacturing (the case of CIMOSA), production management (GRAI-GIM), CIM and other engineering applications (PERA), information systems (Zachman) and Defence architectures management[142] (C4ISR)[143].
Some of the modelling frameworks reviewed are 'complete' in the sense that they can accommodate the complete GERA scope, but each goes into detail from different aspects. Therefore, the user of any of the architectures should consider the GERA scope definition as a guidance or checklist for potential omissions of deliverables that may be needed but are not covered in the respective architecture.
Some of the frameworks have a matrix-like structure, which allows for all possible combinations, while others are more restrictive. **NB** it is often the case that several of the possible combinations offered by such three-dimensional structures may hold content of little practical relevance for most types of modelling tasks. GERA considers this possibility by restricting the scope of some of its aspects (views) to the relevant life cycle phases[144]. Some architec-

[142] 'architectures management' here meaning 'the management of systems architectures in defence'
[143] note that while the origin of these architectures may be in different areas, their contribution is not limited to the industries on the background of which they developed.
[144] e.g., the GERA Identification phase does not exhibit any Human/Machine or Function/Information/etc aspects. This is because in GERA the Identification

tures have all dimensions explicitly represented (e.g. GERA, CIMOSA three-dimensional) while others have implicit dimensions (e.g. PERA, Zachman, GRAI-GIM and C4ISR with respect to the genericity dimension).

This Section has attempted to position the reviewed architectures' modelling frameworks in respect to GERA and may be of help in deciding which combination of frameworks should be used for a particular enterprise modelling task.

The reader should also note that, ultimately, it is the enterprise modelling *methodology* that needs to identify the models that are necessary to support an enterprise architecture endeavour; therefore, the mappings provided in this Chapter is primarily useful for methodology developers and would not necessarily be utilised directly by end users. The methodology developer should consider, based on the objectives of the given change process, which artefacts are necessary and useful – taking into account that practicing enterprise architecture is likely to have multiple objectives, and that artefacts produced should support every one of these.

3.5 Modelling Languages

The practice of Enterprise modelling needs to produce a set of models aimed at an *intended* audience in order to communicate the models' meanings. In order to achieve this purpose, the models must be constructed using languages that (a) are appropriate[145] to the enterprise aspect modelled and (b) are / can be understood by the target audience[146]. For this purpose, existing languages may be adopted (as-is, or customised) or new languages may be created. Any extensions to chosen existing languages must be fully explained and justified, as they in effect change the structure (the metamodel and therefore, implicitly the meaning) of the language itself. Newly designed modelling languages should be based on sound ontologies (metamodels with semantic rules), which will ensure the required consistency and determine the *expressive power* of the modelling as required by the desired enterprise view.

GERAM contains several requirements regarding modelling languages definition (constructs and semantics), expressiveness, required domain coverage of the chosen set of modelling languages and the requirement that models pro-

phase abstracts from decisions that would need the differentiation between these aspects (i.e. this phase is too informal and varied to detail any such aspects).

145 in assessing this, a balance must be struck between the *expressiveness* and the *complexity* of the language.

146 a modelling language contains modelling constructs and rules to combine them (syntax). Therefore, to understand a model one must be familiar with the language used to construct that model.

duced should be able to be integrated across various subject areas of enterprise modelling[147].

3.5.1 Modelling Languages for PERA

PERA does not prescribe any specific language for use in conjunction with its framework, but it does make recommendations regarding the type and necessary features of tools, models and languages to be adopted (Li and Williams , 1994). In essence, the decision to use a particular type of modelling language is either made by the user or by the methodology followed by the user. Some examples of possible language selections (either recommended by PERA or the author of this Chapter) are given in Fig. 3.32.

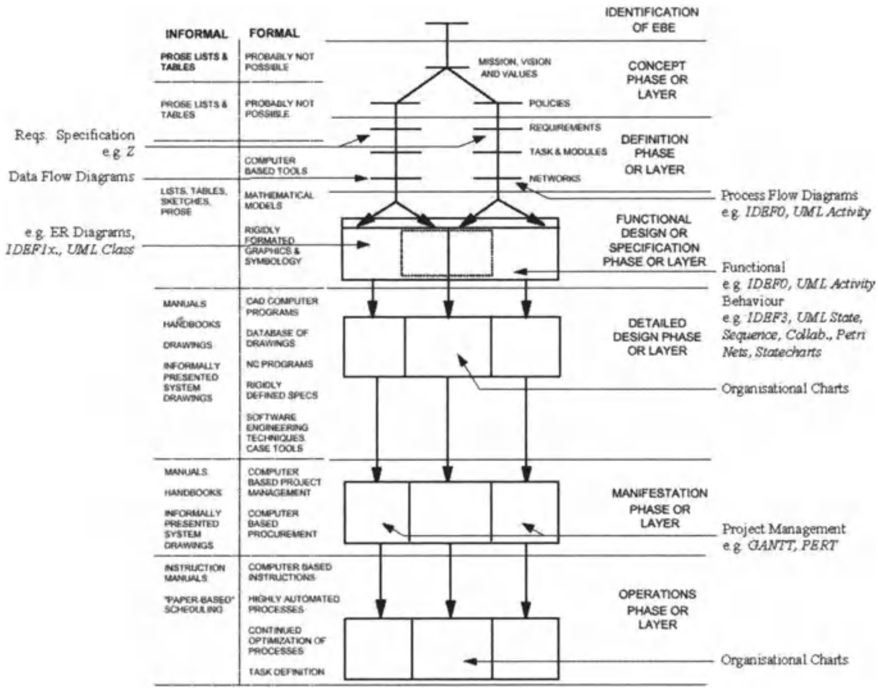


Fig. 3.32. Some proposed PERA tools and languages (based on (Li and Williams , 1994; Williams et al. , 2001))

---

147 the integration in question may be based on a common ontology of the languages and/or using integrity constraints. The need for common (integrated) ontology and metamodel is particularly true in the case of some sets of languages (e.g. IDEF), which are not based on-, or even define a common ontology, despite claiming to belong to the same 'family'.

Notes about Fig. 3.32:

- usable languages / tools which are not explicitly mentioned in the reviewed documents have been shown in italics;
- Data Flow Diagrams may not directly substitute IDEF0 or vice versa[148];
- the set of languages shown is not exhaustive. Alternative or additional languages may be used as long as they meet the requirements set in the reviewed PERA documents.

Also of relevance to modelling constructs is the PERA 'generic task (or activity) module' construct, which is composed of a transformation[149] and a storage. Generic task modules make up functions which may be combined into networks showing information (data) or material / energy flow, depending on the side considered (management and control vs. mission fulfilment) (Li and Williams , 1994).

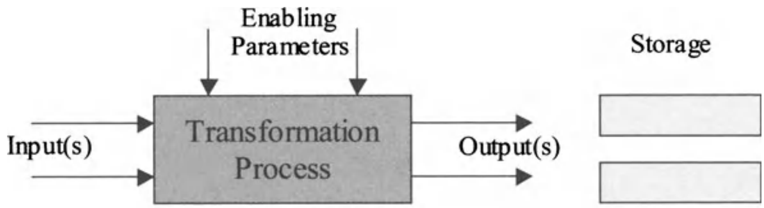


Fig. 3.33. Generic activity module (Williams and Li , 1995)

3.5.2 GRAI-GIM Modelling Languages

The ultimate purpose of enterprise engineering and modelling is *change*[150]. However, a successful change process depends on a deep understanding of the AS-IS state of the system for which the need for change has been identified. If this understanding is not available within the enterprise, then the change process should be preceded by a modelling (or model updating, if the model exists) of the existing system.

148 both languages aim to describe activity models. IDEF0 shows the flow of control and mechanisms while DFDs do not. On the other hand, DFDs explicitly show data stores, which IDEF0 does not. Therefore, the languages are complementary but also somewhat overlapping.

149 similar to the IDEF0 activity construct but without modelling the resources. The transformation process may be e.g. a production process (for the manufacturing tasks) or a written task statement / scope (for the human-based tasks)

150 An 'agile' enterprise needs to (and thrives on) continually change and adapt to its environment.

An essential concept in enterprise engineering is the necessity of user-oriented- and system-oriented phases in the task of deriving the system specifications further used in system design. Ideally, the same people (e.g. the architect(s)) should be involved in both (user- and system-oriented) phases in order to ensure the consistency of the requirements and specifications in the transition from user- to system requirements.
GRAI acknowledges the importance of the AS-IS modelling and the necessity of user- and system oriented design, both in its Structured Approach and in its proposed set of modelling constructs and languages (Doumeingts , 1998a).

	Function	Information	Decision	Physical
Conceptual	Conceptual Functional Model **Actigrams (IDEF0)**	Conceptual Data Model **ER Diagrams**	Conceptual Process Model **GRAI Grid, GRAI Nets**	Conceptual Operational Model **Actigrams (IDEF0) GRAICO, Stock / Resource**
Structural	Structural Functional Model **Actigrams (IDEF0)**	Structural Data Model **Network & Relational models**	Structural Process Model **MERISE Process Model**	Structural Operational Model **Actigrams (IDEF0)**
Realisational	Realisational Functional Model **Computer tool supports**	Realisational Data Model **Data Bases, computer tool supports**	Realisational Process Model **Softwares and computer tool supports**	Realisational Operational Model **"Machines" and Production system physical organisation**

Fig. 3.34. GRAI-GIM Modelling constructs and languages (based on (Chen et al. , 1997))

Notes about Fig. 3.34:

- The Conceptual, Structural, Realisational levels are reflecting the Analysis, User Oriented Design and Technical Oriented Design phases of the GRAI Structured Approach (Chen et al. , 1997);
- The GRAICO formalism was introduced to take the event-driven aspect of the Physical system into account;
- The Stock/Resource model is used (e.g. in preference to an IDEF3 process model) for its ability to express inventory levels and lead-time.
- The Realisational level contains no specific modelling constructs;they are determined on the basis of the tools selected for implementation.

GRAI has made an important contribution to the enterprise integration field, by identifying the inadequate coverage of the decisional aspects of the modelled enterprises and by providing constructs and methodologies to address this issue. One of the main constructs created for the purpose of decisional

modelling is the GRAI Grid, which plays multiple roles within the GRAI architecture framework.
Relevant to this section however is GRAI-Grid's *modelling language* role, used to describe the decisional structure of an enterprise from a managerial perspective.

	function1	function2	function3
horizon1 / period1	DC_{11}(DF,ii,ei)	DC_{21}(DF,ii,ei)	DC_{31}(DF,ii,ei)
horizon2 / period2	DC_{12}(DF,ii,ei)	DC_{22}(DF,ii,ei)	DC_{32}(DF,ii,ei)
horizon3 / period3	DC_{13}(DF,ii,ei)	DC_{23}(DF,ii,ei)	DC_{33}(DF,ii,ei)

DC_{mn}(DF,ii,ei) = (ADJ, ASK, DF, ii)
DF = (Obj, CS, CR, DV)

DC = Decision Centre
DF = Decision Frame (received from upper level or sent to lower levels)
ii = internal information (aggregated from lower levels or sent to upper level)
ei = external information
ADJ = adjust (physical system)
ASK = ask upper level to adjust (if scope beyond current level)
Obj = objectives
CS = constraints
CR = criteria
DV = decision variables

Fig. 3.35. Possible high-level[151]interpretation of the GRAI Grid as a language.

As a modelling language, the GRAI Grid identifies constructs such as decision levels (characterised by time spans - horizons and periods), functions (i.e. 'to manage', 'to plan', 'to control') and decision centres (organisational roles[152] displaying decision-making activities). In this form, the GRAI-Grid could be expressed as shown in Fig. 3.35.
More details on other GRAI-Grid roles are given in (Doumeingts et al. , 1998b) and Section 3.7.2.
GRAI Nets are aimed at a detailed modelling of the activities contained in the decision centres identified in the GRAI-Grid. Based on graph theory, GRAI

[151] some details of the constructs definitions omitted for clarity (e.g. details on the possible geometrical locations of the decision frames (**DF**) given to a decision centre (**DC**), etc).
[152] or organisational units, in GRAI terminology

Nets are primatily composed of *states* (defined by variables and results) connected by *activities* (represented as directed arrows with bullet content, and transforming the inputs in outputs) and supported by *resources* (information or physical). The high-level structure of GRAI Net is shown in the top right-hand side of Fig. 3.36.

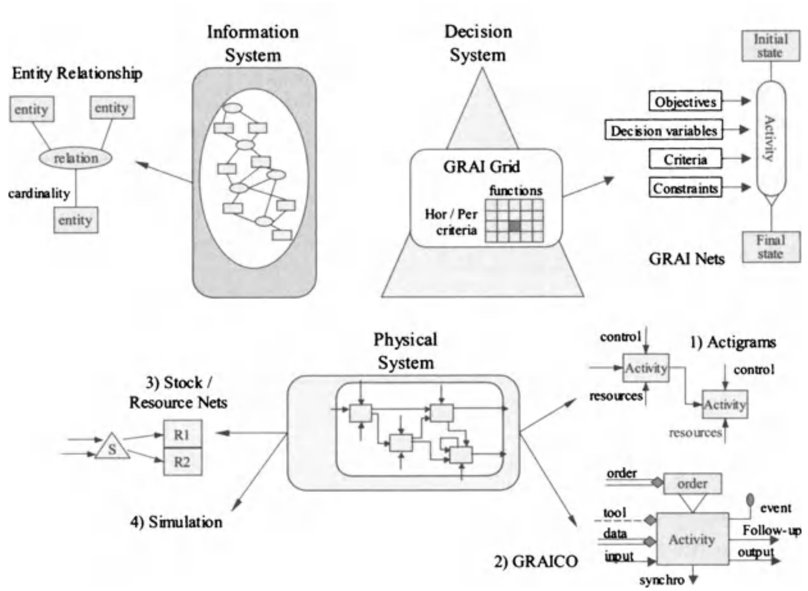


Fig. 3.36. GRAI-GIM Modelling constructs and languages (based on (Doumeingts et al. , 1999))

Refer (Doumeingts et al. , 1992) for more detail about the GRAI modelling languages.

3.5.3 CIMOSA Modelling Languages

CIMOSA views an enterprise as a collection of (possibly concurrent) processes executed by interacting functional entities or agents[153] (Vernadat , 1998). The behaviour of the processes on one hand and functional entities / agents on the other is modelled separately. The reason behind this approach is the CIMOSA conclusion that process behaviour may be defined as a sequence of steps, including ill-defined processes (which may however require additional constructs). The sequence of steps is modelled in the form of a workflow via behavioural rules, which may be extended to include events. Functional

[153] agents may be functional entities pursuing a goal and displaying some autonomy and motivations (e.g. goal-seeking behaviour).

entity behaviour however should be modelled via transitions of states (e.g. using statecharts or similarly expressive languages) rather than a sequence of steps. Hence CIMOSA adopts an event-driven, process based modelling approach, which is reflected in the third-party languages recommended and in the constructs defining the CIMOSA languages (refer Fig. 3.37).

	Behaviour	Function	Information	Resource	Organisation
Requirements Definition	Workflow (WHEN..DO) behavioural rules, *FOL*	Activities, Events, Processes, *IDEF0*	Enterprise Objects, Object Views	Resource, Required Capability Set	Organisation Unit, Organisation Cell
Design Specification	Workflow (WHEN..DO) behavioural rules, *FOL, Z*	As above + time, Petri Nets, (*UML*) Statechart, *IDEF3 OSTD*	Extended ER, *UML Class diagram*	Resource, Complete Capability Set	Organisation Unit, Organisation Cell
Implementaton Description	Workflow (WHEN..DO) behavioural rules, *FOL, Z*	As above + Implemented Functional Operation	Relational model, SQL	As above + Implemented Functional Entity	Organisation Unit, Organisation Cell

Fig. 3.37. CIMOSA Modelling constructs and languages

Similar to other architecture frameworks, CIMOSA identifies several types of *flows* within an enterprise: control (enterprise behaviour), material (physical components) and information (decisions, data, documents).These flows result in a family of Workflow, Functional, Information, Resource and Organisation languages being proposed, which are independent but complementary[154]. Notes relating to Fig. 3.37:

- activities are defined similar to the IDEF0 constructs, with several extensions and additions (e.g. control and resource outputs, activity capabilities)[155];
- enterprise objects may have an identity, associations and properties;
- object views may have identity, nature, properties of Object Set and an Object Set on which the view is defined;
- organisational unit and cell constructs will potentially be updated in the upcoming CIMOSA baseline to support the representation of most types of organisations (matrix, distributed, holonic, etc);
- the resources capability set now includes competencies[156].

[154] for example, a human listed as a resource input in the Function language may have its attributes described in the Information language, its capabilities (now including competencies) described in the Resource language and its responsibilities / authorities expressed in the Organisation language.

[155] refer(Vernadat , 1995) for an in-depth description.

[156] recently, there has been a recognition of the need for the Resource Capability Set to include a Competency construct (Berio and Vernadat , 1999).

- CIMOSA uses Petri Nets and Statechart diagrams to validate its models at the Implementation Description phase[157] (in CIMOSA, simulation, evaluation and validation occur at this phase)

In addition to the above, the Entity Relationship language has been used in describing the CIMOSA major constructs and their relationships[158] ((CIMOSA , 1996; Vernadat , 1998)), which are the building blocks of the CIMOSA metamodel. The CIMOSA Technical Baseline document also mentions IDEF2[159] for dynamics modelling and SADT / IDEF0 and GRAI methodology for use in the functional analysis.

The CIMOSA formal languages approach is well suited for modelling structured processes (e.g. for manufacturing control) but is not so well suited for human behaviour / knowledge modelling (such as in management / decision making). Current efforts within CIMOSA aim at enabling the modelling framework to cope with non-structured activities and processes and allow for social aspects in model simulation. Prime examples are the Competencies construct added to the Capability Set construct of the Resources[160] view and the update of the Organisation Cell- and Organisation Unit constructs of the Organisation view in order to cover a larger range of organisational structures (Berio and Vernadat , 1999).

In conclusion, CIMOSA's Modelling Framework (recently extended to be compatible with the GERA Modelling Framework) is populated with modelling languages (CIMOSA constructs) with the specific purpose of supporting the development of enterprise integration through dynamic creation of model based control systems. If the change process has a different set of objectives, then the CIMOSA user should expect that some additional language constructs would also be needed.

3.5.4 Modelling Languages for the Zachman Framework

In today's ever faster changing business environment, it is understandable that reference architectures aim to remain as generic as possible (a reference). A balance must be reached though between the degree of genericity and the level of usability[161] of an architecture framework. An architecture framework

[157] since the basic behaviour of CIMOSA process models can be translated into Petri nets.

[158] IDEF1x, UML class diagrams or equivalents are possible alternatives to the ER diagrams

[159] IDEF2 is a graphical simulation formalism, unfortunately too closely linked to some particular simulation languages (Bravocco and Yadav , 1985; Vernadat , 1996). For this reason, IDEF2 is fading into oblivion since more generic simulation languages are preferred.

[160] CIMOSA models humans as resources with a specific set of capabilities (resource perspective) , authorities and responsibilities (organisation perspective).

[161] e.g. too general an architecture may not provide any useful guidance as to what deliverables may be needed and how to obtain them, while an excessively detailed

may avoid prescribing specific instances of artefacts (such as a particular modelling language or modelling tool) but still make recommendations or at least describe the essential *requirements* for such artefacts.

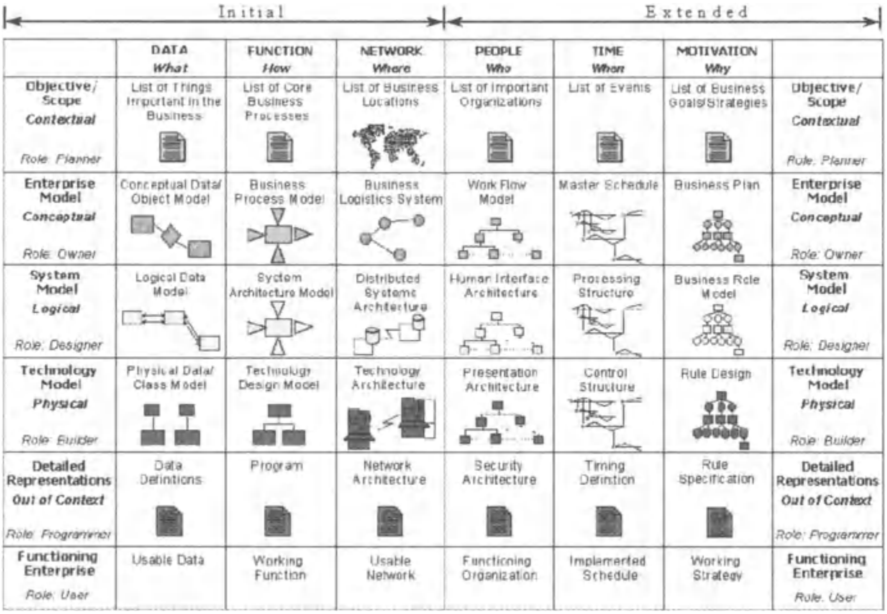


Fig. 3.38. Zachman Framework (Sowa and Zachman , 1992)

In this respect, the Zachman framework does not attempt to define dedicated language constructs (such as e.g. CIMOSA, partially GRAI, etc) nor does it try to enforce any particular third-party modelling languages. The Zachman framework does however suggest[162] several possible modelling languages in its framework cells (refer Fig. 3.38).
Modelling tool / methodologies developers *must* employ various proprietary or third-party modelling languages in order to populate their tools and/or methodologies. The lack of guidance in choosing appropriate modelling languages for the Zachman framework allows users / developers to choose a language they feel appropriate for a specific modelling task, but it does not offer a mechanism to ensure the consistency or interoperability of the models

architecture may constrain the user towards a particular choice of models / tools and even technology (which may become obsolete).

[162] by means of graphical representation (i.e. typical entity and relationship symbols for an ER diagram, map for locations, typical Input, Control, Output, Mechanism (ICOM) symbols for IDEF0, etc. Some recommendations also exist in (Sowa and Zachman , 1992).

created using the chosen languages[163] - unless this task is accomplished by choosing an integrated family of languages, based on a common ontology / metamodel(s).

Figure 3.39 shows one of the many possible selections of languages. This particular selection has been compiled using languages proposed by Popkin Software (Popkin Software , 2001) and additional languages such as Mathematical (First Order) Logic (FOL), Structured English and Decision Tables, which should also be considered as suitable.

	What	How	Where	Who	When	Why
Scope (Contextual)	RP / English	RP / English	RP / Map	RP / English	RP / English	RP / English
Enterprise Model (Conceptual)	ER(M), IDEF1, UML Class	IDEF0, IDEF3 UOB, UML Act, GRAI Nets	Graph	Org Chart, GRAI Grid, GRAI Nets	GANTT / PERT, IDEF3 OSTN, Timed Petri	Struct English
System Model (Logical)	ER, IDEF1x, UML Class	UML Use Case Data Flow Diag.	UML Component	GRAI Grid, UML Use Case	Data Flow, IDEF3 OSTN, Timed / Colored Petri	FOL, Struct English, Z
Technology Model (Physical)	Relational, UML Class	UML Class, Activity, Structure Chart	UML Deployment	UML Real Use Case, UI Design	UML Sequence, Collab, State, Statecharts	FOL, Struct English, Z
Components (Out of Context)	DB Schema	Programming language	URL, IP, TCP/IP	UI Programming language	Struct English	Rule spec. In Prg Lang
Functioning Enterprise	DD L(*SQL*)	Machine code (0/1)	Address, Comm language	User / Worker	English (Schedule)	English

Fig. 3.39. Possible Modelling Languages populating Zachman's Modelling Framework[164].

Notes about Fig. 3.39:

- RP denotes Rich Pictures - using symbols to semi-formally express diverse ideas (concepts). They may be used in combination with text;
- ER(M) denotes ER models allowing M:N relationships (not normalised, which is allowed in the business owner view);
- IDEF1 (rather than e.g. IDEF1x) is proposed at conceptual level to maintain a high degree of independence from the method of implementation;
- GRAI Nets (which normally would belong more to the How abstraction) have also been included in the Who abstraction since they may detail further the activities inside the decision centres.

[163] this is a common problem among the architecture frameworks that offer little or no guidance and language requirements.

[164] proposed and /or alternative languages / constructs are shown in italics.

Moreover, in a broader context Sowa and Zachman (1992) rightfully argue that the Zachman Framework itself may be considered a language[165]. Along the same lines, the models that populate the Zachman framework may be considered the language of the framework[166].

3.5.5 C4ISR Modelling Languages

The C4ISR architecture framework suggests appropriate languages in several areas (C4ISR Architectures Working Group , 1997). Compared e.g. to the Zachman framework, C4ISR specifically proposes some modelling languages, a fact that may improve the interoperability of the models[167] produced. However, the lack of a detailed modelling methodology casts a doubt over the consistency and integratability of the models produced using the proposed languages.

Table 3.1 shows the result of combining the languages proposed by C4ISR with additional languages recommended by Structured Analysis / Object Oriented modelling methodologies (Levis and Wagenhals , 2000),(Bienvenu et al. , 2000) and the author of this Chapter.

For many architectural products, C4ISR describes its own specialised graphical symbols, in effect defining new language constructs. Unfortunately however, these constructs are not accompanied by syntax or semantics (i.e. neither rules of how to combine them, nor metamodels to define the meaning of the constructs are made available). Typically, only a few examples and sometimes a template (which may be considered a (not very specialised) partial model) are supplied. Ill-defined languages must typically rely heavily upon the implicit domain knowledge of the language users. For example, the template for the High level Operational Concept Graphic (OV-1) shows generic shapes of military equipment and empty bullets / connectors, but it supplies minimal information on how these constructs might be used or interpreted.

There are two main situations where this may constitute a problem:

- two different users may model the same domain (or view) of reality using the same language in a different manner and produce a very different model (because there are no syntax rules, such as how many shapes may / must be connected to a connector, if any specific connector must be used on branches, etc). This would compromise the consistency across models produced using the same language;

[165] this concept is applicable to all reviewed modelling frameworks and is in line with the conclusion (stated in Section 3.10.2) that many of the architecture frameworks' components may play a multiple role.

[166] in which case the Zachman framework would become a meta-language / meta-model. This fact supports the hypothesis that the Zachman framework may have an implicit *genericity* dimension. This will be further analysed in future research of the author.

[167] the reader is reminded that models are referred to as 'architectural products' in C4ISR terminology.

Table 3.1. Possible Modelling Languages to support the C4ISR Framework's essential (E) and supporting (S) models

Arch. View	Prod. Ref.	Type	Architecture Product	Languages	Comments
All Views	AV-1	E	Overview and Summary Info	English	Text
All Views	AV-2	E	Integrated Dictionary	*ER / IDEF1x / Text*	ER/ IDEF1x for Dictionary Schema
All views	AV-3	E	Capability Maturity Profile	*CMM / Proprietary*	Proposed
Operational	OV-1	E	High-Level Operational Concept Graphic	*RP / English /* Proprietary graphic	Must contain textual descriptions
Operational	OV-2	E	Operational Node Connectivity Description	Directed Graph	Show connectivity and info flow
Operational	OV-3	E	Operational Information Exchange Matrix	Spreadsheet / Matrix, Relational	Matrix normally not feasible
Operational	OV-4	S	Command Relationships	Organis. Chart, *GRAI Grid*	GRAI-Grid for decisional modelling
Operational	OV-5	S	Activity Model	IDEF0, *UML Activity and Use Case*	Use cases for high level only
Operational	OV-6a	S	Operational Rule Model	Struct Engl, IDEF1x, Decision Tables, Mathematical Logic	Rule modelling. Must be consistent with OV-7, OV-6b
Operational	OV-6b	S	Operational State Transition Description	*UML State, IDEF3 OSTN, Petri Nets,* Harel St. Charts	Behaviour: Dynamics Modelling. Typical to OO.
Technical	OV-6c	S	Operational Event Trace Description	Event Scenario Timing Diag. / *UML Sequence / Collaboration*	Trace sequence of events. Typical to OO. Temporal aspects.
Operational	OV-7	S	Logical Data Model	IDEF1x, *ER*	CADM describes its metamodel
System	SV-1	E	System Interface Description	Graph	Should map to OV-2 nodes

Arch. View	Prod. Ref.	Type	Architecture Product	Languages	Comments
System	SV-2	S	Systems Communication Descript.	Graph	More detail of SV-2
System	SV-3	S	System to System Matrix	Matrix	
System	SV-4	S	Systems Functionality Description	Data Flow Diagrams	Should map to OV-5 activities
System	SV-5	S	Operational Activity to System Function Traceability Matrix	Matrix	Trace how activities of OV-5 map onto functions of SV-4
System	SV-6	S	System Information Exchange Matrix	Spreadsheet / Matrix / Relational	Describes exchanged information
System	SV-7	S	System Performance Parameters Matrix	Spreadsheet / Matrix	Geared to TV-2. Executable evolution
System	SV-8	S	System Evolution Desription	Timeline Graphic	Temporal aspects
System	SV-9	S	System Technology Forecast	Spreadsheet / English	Temporal aspects
System	SV-10a	S	Systems Rules Model	Struct. English, Decision Tables, Mathematical Logic	Rule modelling matching OV-6a (Obsoleted)
System	SV-10b	S	Systems State Transition Description	*UML State Dia, IDEF3 OSTD, Petri Nets*, Harel St. Charts	Behaviour matching OV-6b (Obsoleted)
System	SV-10c	S	Systems Event / Trace Description	Event Scenario Timing Diag. *UML Sequence / Collab. Diags.*	Static view of dynamic behaviour
System	SV-11	S	Physical Data Model	Proprietary / *Relational Model*	Implementation-dependent
Technical	TV-1	E	Technical Architecture Profile	Spreadsheet / Table	Use reference model
Technical	TV-2	S	Standards Technology Forecasts	Spreadsheet / Table	Standards that may apply in the future

- models built using these constructs (such as the OV-1 example provided in the reviewed C4ISR document (C4ISR Architectures Working Group , 1997)) may be interpreted in many different ways by the audience, unless accompanied by comprehensive legends or text clarifying the meanings (semantics) of the symbols[168] used. This may lead to the undesirable situation where the model impedes information sharing[169], rather than facilitating it.

Notes about Table 3.1:

- Languages not mentioned in the original C4ISR document shown in italics;
- IDEF1x used to describe the structure of complex rules (otherwise not suitable for dynamics modelling);
- Command Relationships Chart (OV4) is proposed to be changed to 'Organisational Relationship Chart';
- The Activity Model (OV4) is proposed to become an essential[170] product;
- A new view, the Capability Maturity Profile (AV-3), is proposed for the new C4ISR version.

The C4ISR Core Architecture Data Model (CADM, refer Section 3.8.3.5 and Fig. 3.61) is an example of using a metamodel in C4ISR. This type of representation[171] may also be employed to formally characterize the structure of any of the proprietary languages put forward in some C4ISR views (and thus ensure their consistency and eliminate ambiguities).

3.5.6 ARIS Modelling Languages

ARIS does not prescribe any specific modelling languages, however it does propose several modelling languages in its framework and methodology, as can be seen from Fig. 3.40.

---

[168] all abbreviations or specific terms used *must* be contained in a dictionary (glossary), such as the Defence Data Dictionary System (DDDS, a partial model) or the Integrated Dictionary (AV-2). C4ISR includes a list of abbreviations, which however should be made available (ideally with each model) to the model user(s).

[169] information sharing is mentioned as an important goal in C4ISR. However, *information* as such is simply the interpretation (or meaning) attached to data by humans. Glossaries, metamodels and ontologies used with modelling languages ensure a common interpretation and therefore facilitate *information* -, rather than mere data sharing.

[170] the term 'essential' is also proposed to be changed to 'mandatory' in the next version of C4ISR.

[171] metamodels are typically represented as ER or UML class diagrams.

In the initial phase - Operational Business Problem, ARIS recommends process chain diagrams[172]. As this is a semi-formal phase, Rich Pictures and English (text) may also be used.
The Function, Data, Organisation and Output involve somewhat conventional modelling languages in addition to some proprietary and workflow-specific constructs. In ARIS, modelling of the Function view includes the creation of a decisional model[173]. The Output view appears to be a specialisation of the Data view (the part of the data describing the result of processes). The ARIS Output view is modelled similarly to the Data view, albeit at a lower level of detail[174]. Similarly to the People and Resources abstractions of the Zachman framework (refer Section 3.5.4), the Organisation view does not appear to cover the human aspect of the enterprise beyond the Requirements Definition level, focusing more on the hardware / software aspects instead.
As previously mentioned, the ARIS-specific Control view describes a set relations among models. Therefore, this view will employ either modelling languages that allow hybrid representations, or variations of the existing languages (enriched with the necessary relational constructs). Languages from the first category are e.g. Object-Oriented languages (e.g. class diagrams) where Data and Functions are encapsulated, Colored Petri Nets that allow some attributes to be attached to places, etc. To exemplify the second category (extended languages) , the ARIS Organisation-Data sub-view contains a variation of the organisational chart enriched with the data belonging to each Organisational Unit.
It is typically up to the user to fully understand the purpose of the view in question, take a decision regarding it susefulness for the modelling task at hand and then find a suitable language (from the recommended set, or from elsewhere[175]) to model that view. In this respect GERA may be of help by highlighting the enterprise aspects that need to be covered, which in turn will elicit required models and modelling languages to create the models.
Notes about Fig. 3.40:

- proposed and/or alternative languages / language constructs are shown in italics;
- abbreviations used: DDL for Data Definition Language , DFD for Data Flow Diagram , EER for Extended Entity Relationship, EPCs for Event

[172] the ARIS views are already present at the Operational Business Problem level (which is equivalent to the GERA Concept level), which may be argued as too early a stage.

[173] while the author of this Chapter consider the decisional aspect of enterprises to be a specialisation of the functional aspect, other research directions consider decision as a separate view, or as part of the Organisational view.

[174] the necessary amount of detail is present in the hybrid Output-Data sub-view, part of the Control view

[175] the set of ARIS-recommended languages does not cover all life cycle phases, for all views.

		Initial Situation (Overview)	Requirements Definition	Design Specification	Implementation Description
Function			Structograms Process Seq Processing Forms *IDEF0, GRAI Grid*	Structograms *PETRI Nets* *IDEF3 OSTN*	Programming language
Data			ER, EER Diagrams *IDEF1x* *UML Class*	Relational	DataBase Schema
Output			ER, EER Diagrams *IDEF1x* *UML Class*	Relational	DataBase Schema
Organisation			Organisational Forms / Chart *GRAI Grid mapped*	Network Topology Graphic	Network Protocols
Control	Func - Org	Process Chain Diagrams	Org. Chart with Functions, Matrix, Use Cases	*Org. Chart with Functions, Matrix,* UI Design	Programming Language
Control	Data - Func		UML Class, Seq, EPCs, Statecharts, *Colored PNs, DFD*	UML Class, Compnt Structogram, Trigger Ctrl	OO Programming Language
Control	Func - Outp		UML Class, Activity, Colored PNs, *DFDs*	UML Class	OO Programming Language
Control	Org - Data		Organisational Chart with Data	Authorizations Tables	DDL (*e.g. SQL*)
Control	Org-Output		Organisational Chart with Output	*Tables*	-
Control	Data-Output		UML Class	*UML Class*	-
Control	Org-Func-Data-Output		Process Chain Diags, UML Class	*UML Class*	Programming Language

Fig. 3.40. Potential ARIS modelling languages (based on (Scheer , 1994a), (Scheer and Kruse , 1994c) and (Scheer , 1998b)

Driven Chains, OSTN for Object State Transition Networks and PN for Petri Nets.

NB The formalisms used within the various ARIS views are not fixed. ARIS considers compatibility and view redundancy to be the determining factors in choosing modelling languages (Vernadat , 1996) and as such new modelling approaches are allowed once they have established themselves.

3.5.7 Conclusion About Modelling Languages

Given the diversity of objectives that underlies enterprise integration efforts, practitioners of enterprise architecture need to be able to select modelling languages that allow the expression of their intent and design as needed for the particular task. Some architectures, in line with the GERA Modelling Framework (e.g. PERA, Zachman), aim not to prescribe a complete set of languages, instead arguing that the given engineering domain already has adequate means to describe any deliverable, and the architecture is only there to create a checklist of deliverables and to make their relationships explicit. Such architecture frameworks only give examples of typical modelling languages.
For some areas of modelling, the applicability of modelling languages is of a wide scope. Thus, typical modelling languages may be recommended without restricting the use of the framework to a specific application domain or change objective (e.g. one can generally propose the Entity relationship Data Model for information modelling[176], IDEF0 for activity modelling[177] and IDEF3 for process modelling[178]). On the generic level, (such as shown in the GERA Modelling Framework) this restraint is actually an advantage: the Modelling Framework gains longevity by being non-prescriptive. Therefore, training professionals in its use will provide a long lasting benefit.
There is however a set of common themes in enterprise integration, whereupon most enterprises today share some common objectives likely to persist for a long time. The most important objectives in today's enterprise architecture practice are:

- Dynamic integration of the information and material flow across the supply chain (increasing the level of co-ordination, achieving agility, developing the 'aware enterprise');
- Continuous optimisation of the delivery of service and of the production considering time, production cost and investment;
- Continuous optimisation and management of the use of human resource and knowledge underlying any business endeavour;
- Continuous and dynamic development of the organisation to achieve best organisational fit;
- Continuous optimisation of the deployment and use of technical resources (manufacturing, transport, storage, information processing, communication).

While these objectives are still quite generic in nature, it is possible to identify common needs of change processes regarding types of models or descriptions, such as to express ideas, capture requirements and describe designs. Hence,

176 or alternatively IDEF1x, UML Class diagram.
177 or alternatively Data Flow Diagram, Functional Decomposition.
178 or alternatively Event Driven Process Chain, Petri Nets, UML Sequence / Collaboration diagrams.

some Architecture Frameworks propose a concrete set of modelling languages to (fully or to some extent) populate their Modelling Frameworks. The advantage of setting a standard set of modelling languages is that people can be trained in their use, tools can be developed, and in general the communication of the models and descriptions becomes easier.

Today, no single existing modelling language by itself is capable of modelling all necessary aspects of an enterprise[179]. Therefore, in order to completely and meaningfully[180] model an enterprise, one must presently use several languages. Vernadat (2001) has found an overwhelming number of modelling languages used in many different non-interoperable tools, having an inconsistent vocabulary and based on a weak-, or no ontology at all. The solution currently in development is the proposed Unified Enterprise Modelling Language (UEML[181]), which is to be based on a metamodel and ontology and composed of a set of core and additional constructs. In order to succeed however, such a language must have a large acceptance and appeal to both business users and modelling tools developers. A major benefit of UEML would be a unification of the modelling paradigms terminology.

Separate from the endeavours of modelling language designers, the need to exchange models through the Internet has produced XML, the Extensible Markup Language (W3C , 2000)[182].

As shown in Section 3.5.3, the CIMOSA languages (and its implementations, such as in the FirstStep tool) integrate the language constructs[183] necessary for model-based control. The follow-on UEML language (currently under development) aims to integrate the common modelling constructs, not only for model-based control, but for other objectives as well (such as the ones previously mentioned).

---

[179] such a language would have to be quite sophisticated and therefore hard to use. Rather, a family of strongly bound (e.g. through common ontology), 'leaner' languages covering the same domain would be preferred.

[180] for aspects of the meaning of enterprise models refer (Bernus et al. , 1996a).

[181] in this context, the difference between UML and UEML must be clearly stated. UEML is not an extension or specialisation of UML, but rather a specialised language dedicated to enterprise modelling. UML (as a somewhat general purpose set of modelling languages) may be used to model UEML constructs (Vernadat , 2001). However, any other modelling language(s) may be used provided it has the necessary expressive power - possibly including UEML itself.

[182] XML cannot be considered a true enterprise modelling language, since it is only a means of defining the syntax of documents (and not a means of defining semantics). Thus, one would expect XML Document Type Definitions (DTDs) to emerge that allow the exchange of any model developed in any of the previously mentioned modelling languages. One can also expect tools that can translate such XML documents into an internal representation (such as that of a modelling tool), as well as tools that allow the visualisation of such XML documents.

[183] integrating the constructs of languages means that there is *one* metamodel defining all language constructs, even though for the user, subsets of these constructs appear as separate languages.

The GRAI-GIM Architecture also proposes a set of languages (refer Section 3.5.2), but instead of trying to develop a new integrated language, it selects existing modelling languages that best fit the typical project. While the languages selected are a good match for any of the above listed objectives, their integration as well as the development of semantic foundations (such as envisaged in UEML) would improve the usability of the presently used constructs and will allow better tool support.
Presently there are several families of languages that appear to be suitable candidates for enterprise modelling. Examples:

- the IDEF (ICAM[184] Definition) family of languages. Unfortunately, there is no integrated metamodel available for the complete IDEF language family as yet. As such, consistency between models of the same enterprise developed using different members of the IDEF family cannot be objectively guaranteed (even though some modelling tools supporting the IDEF languages[185] have limited cross checking ability).
- the Unified Modelling Language (UML, (Rumbaugh et al. , 1999)). UML does have an integrated metamodel. In fact many enterprise modelling needs can be supported by UML, however, some crucial ones (organisational modelling, modelling of resources and their capabilities, decisional modelling, separation of required capabilities from actual capabilities, etc) do not have adequate support[186]. The author of this Chapter expects that UEML will bridge this gap.

One should not expect that there will ever be a complete closed set of modelling languages suitable for all projects, once and for all - but one can expect that there will be a reasonably complete integrated set of constructs that support e.g. three quarters of modelling needs for all change processes, with the remaining quarter being selected as needed by the project.

3.6 Methodologies

Many life cycle architectures are accompanied by one or more methodologies aiming to guide the user through the enterprise architecture (including modelling) process. According to Fig. 3.1, GERAM specifies the need for Enterprise Engineering Methodologies (EEMs) which "describe the process of enterprise engineering" (ISO/TC184/SC5/WG1 , 2000a). The coverage of modelling methodologies by the reviewed architectures ranges from general

[184] Integrated Computer Aided Manufacturing, an US Air Force CIM project
[185] typically IDEF0 for functional modelling, IDEF1x for data modelling and IDEF3 for behaviour modelling and possibly simulation
[186] a comparison of the IDFEF and UML modelling languages from the point of view of enterprise modelling is provided in (Noran , 2000b)

guidelines (e.g. C4ISR) to very detailed (representing a true backbone of the architecture itself[187], e.g. PERA).
GERA sets several requirements for modelling methodologies, such as:

- the need to cover human role (the need to involve users in the analysis and design phases);
- the necessity to distinguish between user oriented- and technology oriented design;
- the requirement for the methodologies to use project management techniques;
- the need for an economic aspect (e.g. allow cost and performance evaluations and comparisons).

Few methodologies cover the full set of requirements in the GERA sense. Therefore, it is often necessary to adopt a *set of* methodologies, rather than one single methodology. Overlapping coverage of the selected methodologies may then be used (if necessary) towards the purpose of triangulation[188].
A comprehensive review of all commercial, proprietary modelling methodologies based on the publicly available reviewed architectures is beyond the scope of this Chapter.

3.6.1 The PERA Enterprise Engineering Methodology

The main component of PERA is the Purdue Methodology, published as a 'Handbook for Master Planning' (Williams et al. , 2001). The PERA diagram was initially intended as a graphical guide (or a high-level model) to accompany the methodology, but as shown in Section 3.4.1 it also plays the role of a reference architecture (since it specifies the scope of the necessary deliverables at each life cycle phase)
The PERA diagram describing the enterprise integration methodology is shown in the right-hand side of Fig. 3.2. The Purdue Methodology is a guide for the implementation of Master Planning. The methodology covers the identification, concept, requirements and preliminary design phases, and gives detailed advice on both technical and organisational / human relationship aspects of such projects.
The Master Plan (or Architectural Plan) is the result of the Preliminary Design phase. A distinct characteristic of the Master Plan is that it allows management to make informed decisions about the feasibility of the plan's implementation, as well as of its cost and the time needed to implement it.

[187] the reader is reminded that life cycle architectures attempt to describe the entire enterprise integration endeavour or part of it, rather than providing a snapshot structure of the whole- or part of the enterprise at a given point in time. As such, life cycle architectures themselves are a high-level *model of the methodology* to integrate an enterprise.

[188] the application and combination of several research methodologies in the study of the same phenomenon - for example, using the principle outlined in (Jick , 1979).

Master planning has several advantages:

- the cost of production of a Master Plan is small relative to the cost of detailed design and implementation, thus it can be used for decision support and bidding such as for large engineering projects or for a longer term enterprise integration programme;
- a Master Plan does not have to be implemented in one stage. Depending on the available capabilities, a Master Plan may form the basis of a strategic enterprise integration programme, implemented throughout several years;
- the Master Plan defines sub-architectures of the entity in question, such as the human-organisational architecture, the production and service delivery architecture, and the management information, communication and control system. These systems may be implemented by separate (and possibly parallel) projects (each as one effort, or in stages), thus decreasing the complexity (and therefore risk) of each of these projects. These separate projects would use domain specific methodologies, such as facility engineering, software engineering, and organisational design. The coherence of these separate projects is ensured through the interfaces defined in the Master Plan.

For further details on master planning refer (Williams , 1994b) and (Williams et al. , 2001). The PERA methodology addresses most of the requirements set out in GERA. The PERA Methodology has been successfully used in many industrial projects[189].

3.6.2 GRAI Methodologies

As shown in Section 3.5.2, GRAI recognizes the necessity to model the existing state of an entity before attempting to devise a process for its change and recommends different sets of languages / constructs for the two modelling phases. As a consequence of this approach, GRAI provides a set of modelling methods which together meet most of the GERA requirements. Some of these methodologies are briefly presented in the following.
BenchGRAI is a benchmarking structured approach for bid preparation (based on GIM in respect to the formation and composition of the focus groups, shown in Fig. 3.41). BenchGRAI comprises several phases:

- the modelling of the bidding process (using IDEF0);
- definition of the objectives, such as Bid Preparation Process (BPP) Leadtime, BPP cost conformity and BPP lead-time conformity;
- evaluation of the internal- (within the same enterprise) and external (between various enterprises) results in respect to the objectives.

As a result of the last phase, a 'best practice' approach may be built by selecting the best-performing candidate bidding processes.

[189] notably Flour Daniel, with the company using this methodology for all of its large (billion dollar) engineering projects (Rathwell and Williams , 1996).

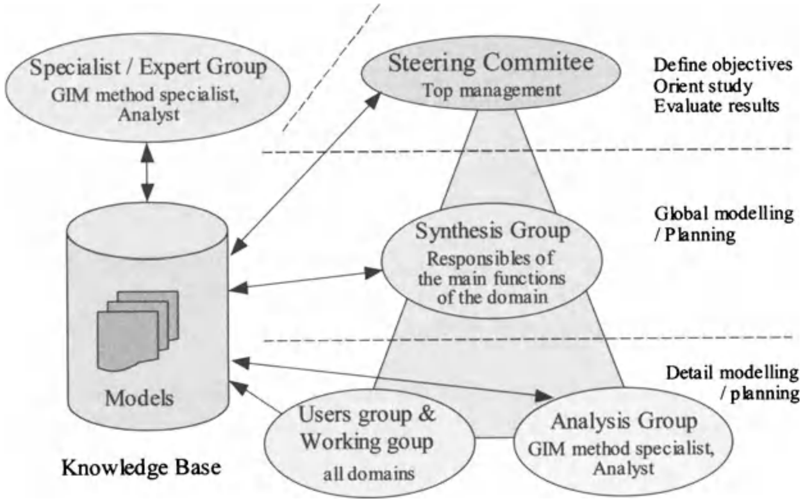


Fig. 3.41. The GRAI groups approach (Doumeingts , 1998a; Chen et al. , 2002)

For the management side of an enterprise to be effective, it needs feedback - i.e. it must know the effect of its actions on the enterprise operations. This may be achieved using performance indicators. EcoGRAI is a method proposed by GRAI to design and implement a system of performance indicators. EcoGRAI comprises several phases:

- modelling of the enterprise control system (using GRAI-Grids and GRAI Nets);
- identification of the global objectives and decision centres' objectives (called drivers) and any conflicts between the drivers and global objectives;
- design of the performance indicators;
- integration of the indicators in the Production Information System.

ECOGRAI also uses the GIM structured approach concept of focus groups.
The GRAI Evolution Methodology (GEM) models the continuous development of enterprises by modelling AS-IS and SHOULD-BE states. As shown in Section **3.3.3** and Fig. 3.13, GEM uses so-called 'NEXT STEP' (shorter term) phases in order to meet the need to readjust the SHOULD-BE state periodically. Each NEXT STEP state reached, then becomes the current AS-IS state. GEM is described in more detail in (Doumeingts , 1998a).
The GRAI Integrated Methodology (GIM) proposes the Structured Approach shown in the right-hand side of Fig. 3.3. Specific to GIM (and important from

the GERA human aspect) is the involvement of specific groups (refer Fig. 3.41)[190]:

- a project board (top management), which supervises and may alter or halt the modelling process[191].
- the responsible personnel from customer service / manufacturing (the synthesis group), which are able to follow, guide and validate the modelling process using their technical skills;
- an analysis group, which has to (a) collect necessary relevant data from the modelled enterprise (analysts) and (b) give advice to the synthesis group to ensure consistency with the GIM method analysis group (GIM expert(s));
- the users group, which provides the necessary information to the other groups.

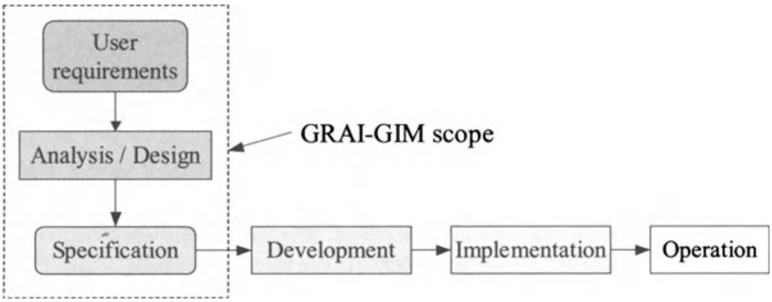


Fig. 3.42. Scope of GRAI-GIM in relation to the life cycle of a system (Chen and Doumeingts , 1996)

The phases in the modelling methodology closely follow the phases present in the GRAI reference architecture and have been described in Section 3.2.2. Essentially, the GRAI-GIM Structured Approach provides advice on the tasks involved in the user- and system requirements definition and the preliminary (architectural) design phases.
Other methodologies have been developed within GRAI, such as GRAI Message (elaboration of business plan), GIMSOFT (specifications for choosing / implementing ERP / Production Management software packages) and GRAI Engineering, for controlling the design of products (Doumeingts et al. , 1999).

[190] the GIM Structured Approach explicitly addresses GERA's 'human factor' methodology requirement.

[191] GRAI acknowledges herewith the need to involve and secure management support for the change process.

Similar to the Purdue Methodology however, none of the GRAI methodologies covers the GERA Implementation, Operation or Decommission phases[192] (refer Fig. 3.42).

3.6.3 The CIMOSA Methodology

CIMOSA provides a Business Modelling Process (Zelm et al. , 1995) based on the CIMOSA specifications contained in (CIMOSA , 1996). Since CIMOSA understands enterprise modelling in terms of processes, activities and operations, it is expected that a CIMOSA modelling methodology would provide guidelines regarding the creation and execution of CIMOSA compliant enterprise engineering processes. These guidelines include (Vlietstra , 1996):

- identification of the business domain;
- building block selection from a catalogue (i.e. selection of partial models);
- customisation and specialisation of selected building blocks, resulting in the particular enterprise model;
- release of the enterprise model into the operational environment;
- add / instantiate variables at execution time (on same or various copies of the released model).

The CIMOSA-defined set of processes involved in enterprise engineering follows the CIMOSA life cycle phases (Requirements Definition, Design Specification, Implementation Description) but it includes an additional Model Maintenance task (a repeatable process, triggered e.g. by change request events originated by customers, new technology, change in constraints, etc). The CIMOSA methodology does not prescribe an order in the life cycle phases involved, allowing for top-down-, bottom-up- or iterative approaches.
The modelling methodology[193] proposed by CIMOSA does not include a detailed coverage of the human and organisational aspects of enterprise engineering projects. While modelling responsibilities and authorities (recently also allowing for human skills / competencies), the methodology does not cover issues such as:

- persuading the management of the need for change and securing its support for the change process;
- policies and procedures needed to support enterprise engineering projects;
- taking into account workforce characteristics, union rules, the role of the champion and the initiating sponsor;
- assessing the cultural and social factors in the enterprise, etc.

Therefore, users of the CIMOSA methodology must complement their methodological knowledge and preparedness with other methodologies, such as the possibility of using the PERA Methodology (to guide the organisation of the

[192] these phases are considered to be covered by domain-specific methodologies.
[193] often referred to in CIMOSA as the 'model derivation process'.

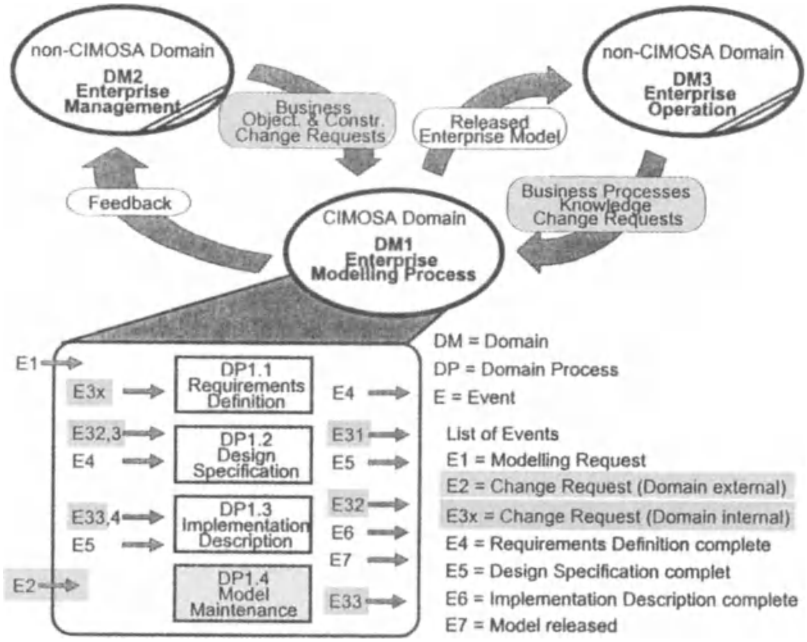


Fig. 3.43. The CIMOSA Modelling Process (based on the description in (Kosanke and Vernadat , 1998)).

Master Planning effort) and a Software Engineering methodology (to guide the organisation of software development projects).

3.6.4 Zachman Methodologies

The Zachman framework aims to be generic by not prescribing any implementation-, or modelling methodology. However, some high-level guidance is still available. Similar to other frameworks[194], Zachman identifies[195] two main phases in enterprise modelling:

- model the existing enterprise so as to improve existing operational processes;
- change the enterprise using a generalisation of the models developed.

[194] refer Section 3.5.2 for the GRAI example.
[195] on the Zachman Institute for Framework Advancement (ZIFA) web site, at the time of writing (Zachman , 2000b).

In the first phase, the framework is populated with particular models of the existing enterprise. The second phase employs a formalisation and generalisation of the particular models in a bottom-up approach, in order to effectively obtain partial models[196].
As it is often the case, various proprietary modelling methodologies relating to the Zachman modelling framework have emerged. Among these are ForeSight, developed by the Zachman modelling framework's authors, the Popkin Process (Popkin Software , 2001), the Visible methodology and Ptech's Causal Architecture[197].
The Popkin process offers a methodology to identify content for the Zachman framework cells, to relate the cells and to choose the appropriate languages / tools to achieve these goals. The Popkin process methodology also provides high-level guidance for choosing a suitable, out-of-the-box 'established' methodology (Business-, Data-, Structured- or Object oriented modelling).
The Ptech Causal Architecture approach uses Causal Loop Diagrams[198] (CLD) models to identify relevant content for the Zachman framework cells. CLDs are used in combination with value- and causal mappings (Vail , 2001) to identify key values and high leverage factors for the enterprise. The Causal Architecture methodology then provides a guidance to map these values to the Zachman cells and assign priorities to them.
A complete review of all significant Zachman-based methodologies is beyond the scope of this Chapter.

3.6.5 C4ISR Modelling Methodologies

The C4ISR architecture framework supplies only high-level modelling directions (shown in Fig. 3.44) but also describes a set of mandatory / supporting products (as shown in the Product Reference and Architecture Product columns of Table 3.1) expected to result from an architecture process. C4ISR does not explicitly describe a detailed process for developing views and products, or the organisation of architecture development projects. Generally, the absence of a modelling process may significantly impede upon the integratability and interoperability of architectures produced using C4ISR.
However, as shown in Section 3.2.5, it is possible to relate the GERA life cycle phases to the C4ISR views (as shown in Fig. 3.6) and high-level modelling phases (also described in Fig. 3.8). Also, the top-down view descriptions in Table 3.1 (based on (C4ISR Architectures Working Group , 1997)) describe in some detail the architecture deliverables, hence giving some guidance to the user regarding the progression of enterprise architecture projects.

[196] refer Section 3.7 for more information on partial (or reference) models.
[197] these methodologies are not described here in more detail due to their proprietary nature. More information may be obtained from the authors' web sites - at the time of this writing http://www.zifa.com , http://www.popkin.com, http://www.visible.com and http://www.ptech.com.
[198] a Systems Engineering approach used to describe cause / effect narratives.

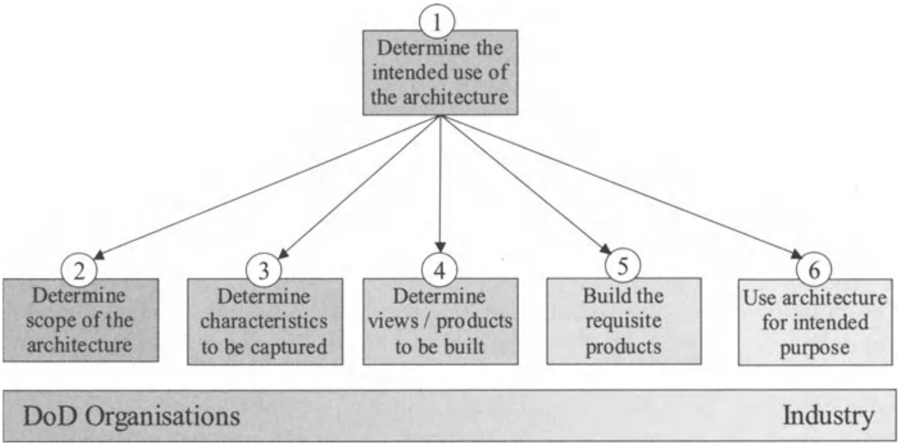


Fig. 3.44. Guidance provided by C4ISR (C4ISR Architectures Working Group , 1997)

Proprietary methodologies have also emerged, such as META Group's Enterprise Architecture Strategy (EAS) applicable to C4ISR. The META Group defines a 5-step process which progressively (and iteratively) builds the enterprise business, information, technical and application portfolio architectures, followed by guidance regarding gap analysis and migration / implementation planning (META Group , 2002).
Popkin Software also provides a dedicated module in its proprietary tool (System Architect) and methodology support for modelling the C4ISR architectural products[199].
Other, publicly accessible proposed C4ISR modelling methodologies reviewed follow two main directions:

- Structured Analysis and Development techniques, notably the six-stage process shown in (Levis and Wagenhals , 2000)[200] and (Wagenhals et al. , 2000);
- Object-oriented paradigm - such as the Object-oriented method described in (Bienvenu et al. , 2000) and (Levis and Wagenhals , 2001b)[201].

[199] as previously mentioned, such methodologies cannot be discussed in detail due to their proprietary nature. The reader is directed to their web sites for details. At the time of this writing, these web sites were http://www.metagroup.com and http://www.popkin.com.

[200] this approach uses ER to produce a partial ordering of the deliverables from which a series of steps (and hence a methodology) may be obtained (Levis and Wagenhals , 2000).

[201] it must be noted here that the Object Oriented method listed builds on the processes described in the Structured Analysis modelling methodology, rather

Unfortunately however, such methodologies do not cover some essential technical elements necessary for enterprise architecture development in the GERA sense, such as the optimisation of human *versus* technical tasks[202], and are usually vague of the development of the human-computer interface[203]. The design of human tasks is another aspect not covered by these methodologies. Therefore, C4ISR users are advised to select a suitable *suite of* methodologies[204] covering the missing elements, as noted above and defined in the GERAM requirements specification.

3.6.6 The ARIS Methodology

The ARIS House framework (described in Section 3.4.6) may be used to guide the ARIS modelling effort (Scheer , 1998b). However, a more complete methodology model may be obtained by combining the ARIS House and the ARIS HOBE (also described in Section 3.4.6, Fig. 3.31) frameworks into a three-dimensional structure, as shown in a simplified form in Fig. 3.45. Starting with the ARIS HOBE Process Engineering (Level I), each view in the ARIS House (Function, Data, Output, Organisation and Control) is modelled in turn, for each ARIS life cycle phase, with emphasis on the Requirements definition life cycle phase. Populating the ARIS HOBE Level I with models enables the subsequent modelling of the other ARIS HOBE levels (II to IV[205]).
The modelling methodology covers entities identified in Fig. 3.29 within the ARIS views. Thus, for example modelling of the function view covers functions, goals (i.e. the business rules) but interestingly also decisional aspects[206]. It is also worth noting that some entities are common to multiple views (e.g. information services being part of Output, but also part of Data).
The ARIS views are modelled in all life cycle phases with the exception of the Output and Control. The Output view is modelled only at the Requirements Definition level[207].

---

than being an *alternative* to Structured Analysis. This is generally true of most Object Oriented methodologies and languages (e.g. the UML diagrams).

202 i.e. the 'extent of automation' in GERA / PERA terms.

203 hence, it is highly recommended to extend such methodologies with a Human Computer Interface design methodology.

204 e.g. PERA's automation extent concept for human vs. non-human task modelling and GRAI's Grid for decisional modelling.

205 ARIS HOBE Level V is considered to be the framework itself, as previously shown in Fig. 3.31.

206 Often, decisional aspects are covered together with organisational aspects. In the author's opinion, decisions are a special kind of functions, while the organisation actually presents the mapping of the resources onto the functions ('who does what').

207 this may be due to the fact that ARIS is mainly concerned with the development of the automated parts of the enterprise

As previously mentioned, the Control view in ARIS serves the purpose of bringing together the other views and highlighting their relationships. Therefore, modelling of the Control view must include all bilateral relationships of the views, which are modelled in ARIS at various life cycle depths, as follows. All bilateral relationships are modelled in the requirements definition life cycle phase. Some (Functions-Organisation, Functions-Data and Organisation-Data) are modelled down to the Design Specification level, with only the (generally best-understood) Functions-Data relationship modelled down to the Implementation Description level.
In conclusion, the ARIS-proposed modelling methodology recommends a sequence in modelling the ARIS House life cycles and the HOBE levels[208].

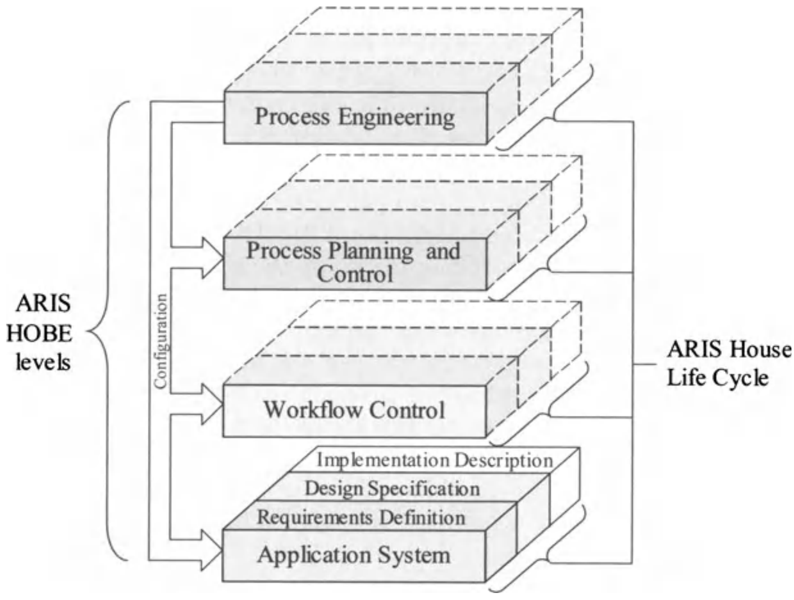


Fig. 3.45. Simplified ARIS methodology model and modelling framework (based on (Scheer , 1998b))

The modelling methodology employed by the ARIS architecture framework is described in detail in (Scheer , 1998b), which describes the modelling of all individual views, plus the Control view (containing the bilateral view relations and a comprehensive view). Again, the human role coverage requirement as defined in GERA is not fully met - therefore, it is recommended than other suitable methodologies should be employed to complement ARIS.

[208] in the case of HOBE, the succession is suggested by the left-hand side composite arrow shown in Fig. 3.45.

3.6.7 Conclusion About Methodologies

As shown in Section 3.2, the life cycle phases contained in the reviewed life cycle architectures do not include any succession or other temporal aspects. However, modelling methodologies would often prescribe a certain selection and/or succession of life cycle phases (i.e. *suggest* a particular life history) for a given (or a given type of) modelling task.
Apart from the PERA methodology, partly GRAI-GIM (which addresses the project organisation) and partly the META EAS process, all other methodologies concentrate on the technical tasks of model / deliverable development. A possible reason for this is that consulting companies have the expertise in the organisational aspects of architecture development, and since these elements are so essential for success, they are able to hold the market captive. In addition, proprietary methodologies are usually difficult to assess due to the lack of publicly available details. The proprietary nature of such methodologies may provide short-term market gains, but could affect their general acceptance in the long term.
Companies need to develop in-house capability to exercise enterprise architecture at least to the extent that they should be capable of making informed decisions about the time when external help is needed in the process and about the *type* of help required.

3.7 Reference Models

Presently, knowledge management (including its preservation[209]) appears to be one of the main drivers of enterprise modelling. A 'good' enterprise model is able to encapsulate (and therefore preserve) knowledge that would otherwise be lost[210]. Of course, knowledge preservation makes most sense in view of its (entire or partial) *reuse*. In enterprise modelling, reuse may be achieved by means of Reference- or Partial Models[211], displaying various levels of specialisation (and hence various degrees of applicability[212]). For a reference architecture, partial models take the form of solutions for a particular *type* of modelling problem. They are a potential resource saver for the enterprise modeller.
Partial models in the GERA sense may be prototypes, abstract models or models of classes of enterprises which must be specialised and ultimately instantiated in order to obtain the model of a particular enterprise. GERA specifies human role (organisation, responsibilities, etc), process and technology

[209] which is often overlooked in the most relevant life cycle phase: the decommissioning of enterprises and resources.
[210] e.g. the tacit process knowledge owned by a retiring senior employee.
[211] 'reference models' and 'partial models' are used interchangeably in this Chapter.
[212] the more specialised a partial model is, the narrower the area of its potential application becomes.

as possible domains for the partial models. Emphasis is being laid on partial models covering technology-oriented IT systems and Integrating Services, which greatly assist in the enterprise engineering effort[213].
The reader should also be aware at this point that architectures describe structure at a given level of abstraction - e.g. partial models may exist on the policy level (Concept life cycle phase), requirements level, architecture level, etc – even including the detailed design level (where partial models might describe various views of an often used actual product).
Therefore, in a broader context *architectures themselves* may also be considered partial models. This is in line with the finding that the components of architecture frameworks may play multiple roles, resulting in the distinction between such components often being blurred [214].

3.7.1 A PERA Partial Model

PERA contains a partial model for the management and control side (the CIM Reference Model), described in (Williams , 1988) and subsequently adopted as a standard by the Instrument Society of America (Instrument Society of America, (ISA , 2000)).
The Purdue CIM Reference presents the Information Functional Network and the Information Systems Architecture of the Purdue Architecture (refer Fig. 3.46), without considerations of system construction and operation. It refers to the automatable functions of an enterprise, while defining humans and human-related functions as 'external entities'. The exchange of information between worker and plant is enabled via communication facilities (Li and Williams , 1994).
The CIM Reference Model presents a partially specialised operational structure of an integrated enterprise and is therefore usable for the development of the deliverables included in the PERA Master Plan[215].
The PERA CIM reference model maps onto the partial level of GERA as shown in Fig. 3.46(the shaded areas).
Other PERA examples are also available for various industries - notably the Fluor-Daniel example presented in (Rathwell and Williams , 1996). Depending on the degree of detail provided and referring to the GERA guidelines, such 'examples' may be considered templates, 'class models' or patterns[216]. They may be reused by:

- 'filling in' the details, in the case of templates;

[213] this type of partial models are similar for most enterprises and thus may be widely reused
[214] refer Section 3.10.2 for a more detailed description of this aspect.
[215] The PERA Master Plan is the result of the Preliminary Design phase.
[216] patterns: robust solutions to frequently occurring problems, consisting of name, problem description, solution description and possibly consequences / tradeoffs.

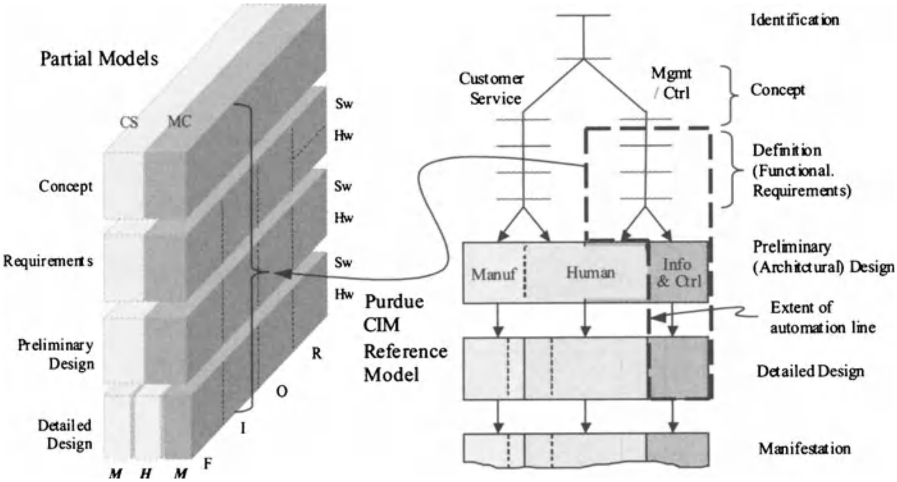


Fig. 3.46. The Purdue CIM Reference Model mapping onto the PERA (Li and Williams , 1994) and the GERA

- further specialisation and subsequent instantiation (into a particular model), in the case of class models;
- comparison of the 'problem' component with the modelling task at hand in the case of patterns. If a close enough match is found, the 'solution' component of the matched pattern is customised (using any available 'trade-off' components of the pattern) and then adopted.

3.7.2 Partial Models in GRAI-GIM

Reference models may exist at various levels of detail, depending on their purpose and intended audience. As with any model, a balance must be reached between model complexity and expressiveness.
The GRAI architecture framework is based on the control and systems theory, according to which all systems are composed of a control- and a physical subsystem. The control system makes decisions by using external information and models of the physical system (which must be kept accurate by means of feedback or internal information). The system containing the external / internal information and its flow makes up a third (information) subsystem.
In line with the above, the graphic in Fig. 3.47 may be considered a high-level partial model of the GRAI concept. Its complexity is low but so is its expressiveness (e.g. it cannot show the Management, Command and Control System structure or the relation between its components).

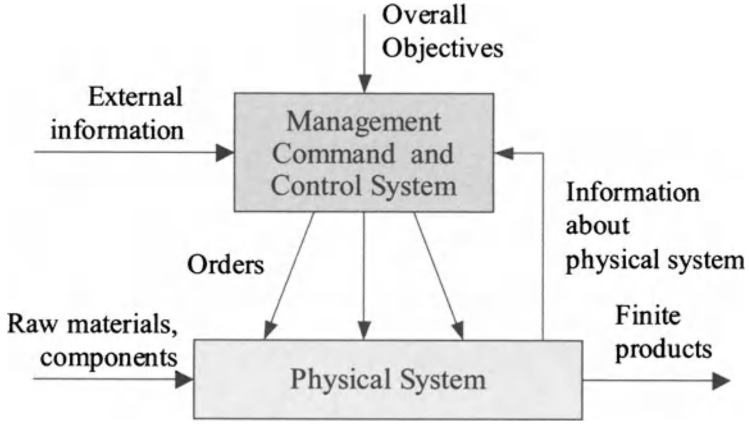


Fig. 3.47. A GRAI-GIM reference model (Chen and Doumeingts , 1996)

This reference model is usable e.g. in the GRAI-GIM Definition (GERA Concept) life cycle phase, or when attempting to gain upper management support for a change process and too much detail would actually impede communication. A simple model of the enterprise may be constructed by instantiating the partial model shown in Fig. 3.47.
The reference model shown in Fig. 3.48 (the GRAI conceptual model) may be derived (specialised) from the model in Fig. 3.47. This model is more complex, but also more expressive and thus suitable for e.g. the GRAI user-oriented design life cycle phase.
In GRAI terminology, the Decision system represents the management side of the enterprise, for which a dedicated partial model (the GRAI-Grid) is available. The Operating System refers to the control functions of the enterprise, while the Physical system represents the enterprise operations[217].
The GRAI Grid is a partial model for the decisional view of the GRAI architecture framework, at the requirements level (Doumeingts , 1984). It is derived (specialised) from the GRAI conceptual model and as such it enables the user to model constructs inherited from the GRAI model (such as decision centres, decision levels, internal and external information). In addition to its parent partial model however (Fig. 3.48), the GRAI Grid also describes types of decisional functions, decision frames and in addition it further details decision *centres* (presented in a GRAI Grid as intersections of decision functions and levels) (refer Fig. 3.49).

[217] together, the GRAI Decision and Operating Systems would map onto the Management and Control part of the GERA (or PERA) while the GRAI Physical System would map onto the GERA (or PERA) Customer Service part.

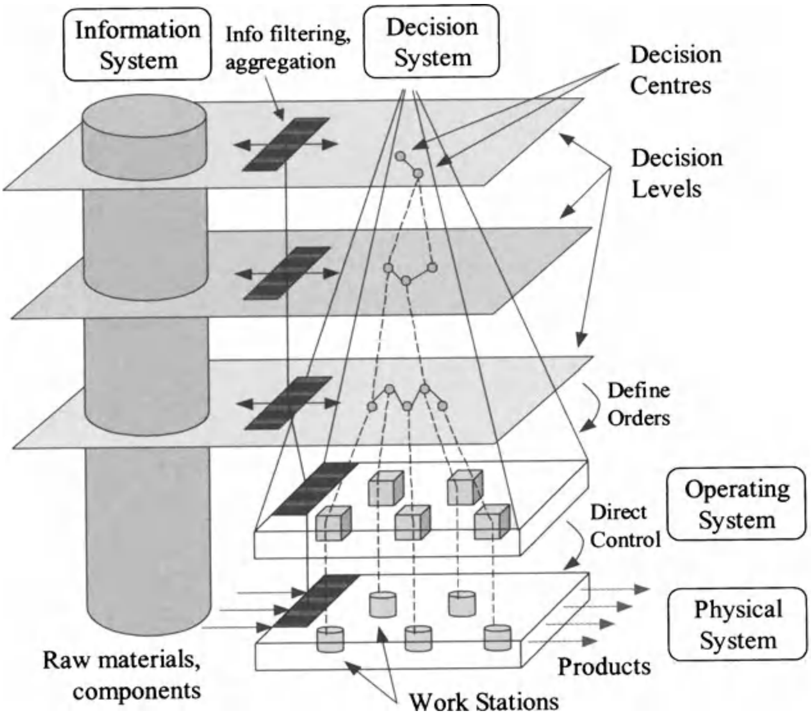


Fig. 3.48. GRAI-GIM Conceptual Model (based on (Doumeingts et al. , 1992))

It is also possible to describe the functions of each decisional centre present in a GRAI Grid by using an activity modelling language such as IDEF0 or GRAI Nets. Moreover, once the GRAI-Grid is complete, it is possible to map the (human and non-human) resources on the GRAI-Grid, therefore obtaining the organisational structure of the enterprise. This makes the GRAI Grid an important tool in the partial modelling of the functional, resource and organisational aspects of an enterprise[218]. The GRAI-Grid may thus be used to identify potential serious problems in an organisation, such as over-managed hierarchies, conflicting roles, narrow- or paternalistic management, etc.
As a 'coarse' partial model of an enterprise, the GRAI-Grid maps onto the GERA Partial level on the Functional, Organisational and possibly Resources views at the Requirements life cycle phase, on the Management and Control side as graphically shown in Fig. 3.49. This happens because the decisional aspect depicted by the GRAI-Grid is first and foremost functional, while also

[218] **NB** separate functional and resources modelling are still necessary for the enterprise in question.

involving the organisation and some of the resources[219].

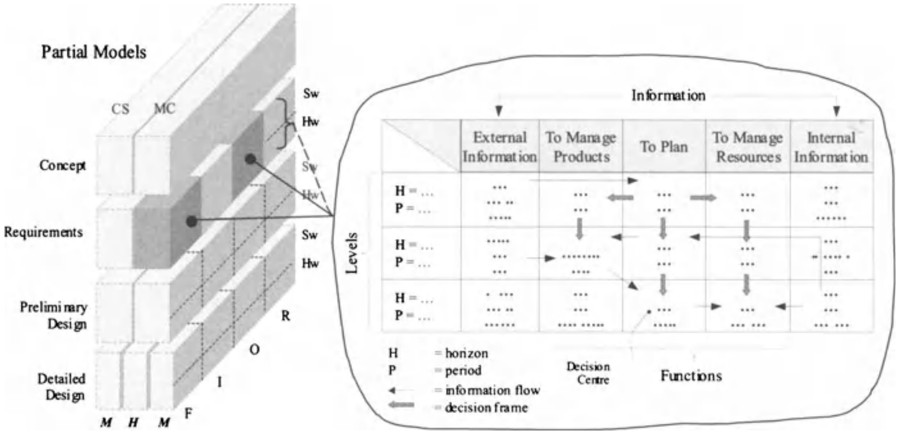


Fig. 3.49. Mapping the GRAI-Grid (Doumeingts , 1984) onto the GERA partial level[220].

Typically for partial models, the GRAI conceptual model (and the GRAI-Grid as its specialisation) makes use of generic GRAI concepts such as decision centre[221], decision level, horizon, period, etc. According to GERAM, these concepts should also be *formally* described in a metamodel and ontologies (such as the control and systems theories that GRAI is based on).
For more information about the GRAI conceptual model and the use of the GRAI-Grid in the GRAI architecture framework refer (Doumeingts et al. , 1998b).

3.7.3 A CIMOSA Partial Model

Information Technology (as an enabler of Information Systems, which in its turn is a subsystem of the business) and Manufacturing Technology (as an important enabler of enterprise operations) still present a high degree of heterogeneity, both in hardware and software aspects. Thus there is a need for an additional layer between the business user and the enabling technologies which would hide this heterogeneity, facilitate the information exchange / sharing

[219] some resources may also appear e.g. as mechanisms in the IDEF0 models of the decision centres.

[220] A possible language form of the GRAI-Grid has been shown in Fig. 3.35

[221] the decision centre itself is described by a separate reference model (Chen and Doumeingts , 1996).

and provide a 'plug and play' environment for components[222] (Vernadat , 1996).
Executable models (such as those that CIMOSA endeavours to create) have to exist at an abstraction level understood by the envisaged execution infrastructure. Hence, the execution infrastructure capabilities may affect the necessary degree of formality and detail of the model[223]. In CIMOSA, the integrating infrastructure is used both in the model engineering (maintenance)- and the model execution phases, in order to ensure modelling consistency and to minimise the impact upon model deployment. This is reflected in the CIMOSA Integrating Infrastructure Services (IIS) structure shown in Fig. 3.50.

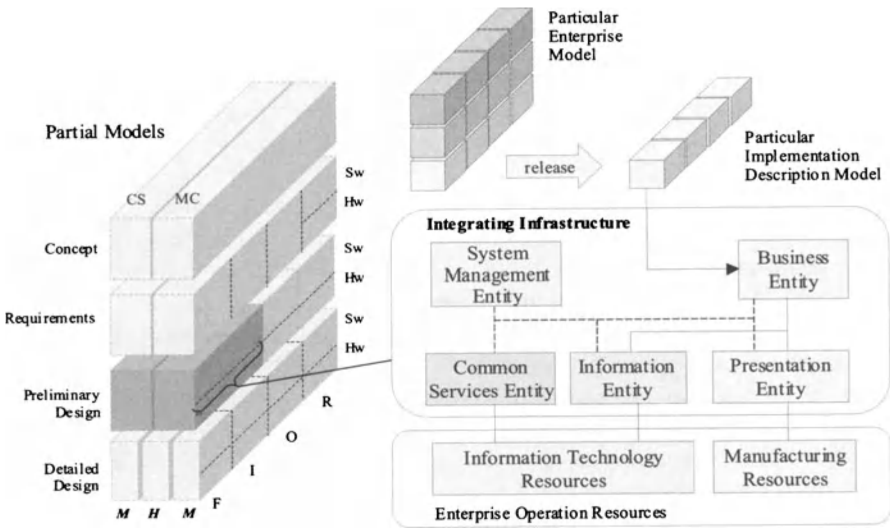


Fig. 3.50. The specification of the CIMOSA Integrated Infrastructure (AMICE , 1993) and its mapping onto the GERA Partial Level

The IIS may be specified as a partial model in the sense that it only defines generic service entities (only its functionality is defined), its implementation being at the discretion of the IT vendors. The IIS provides services for func-

222 this additional layer is frequently referred to as 'middleware'.
223 the more 'knowledgeable' the execution infrastructure is, the less detailed (and less formal, if the basic rules are built-in) the model needs to be. Simple example: computer hardware components with built-in (software) processing capabilities need significantly less external software instructions to accomplish the same amount of computing. In this case, most of the elementary instructions are already 'known' by the hardware. The same applies for the human side: the amount of 'software' instructions needed for a human varies greatly with the human's implicit knowledge (the built-in capability ('software') of the 'human hardware').

tion, information and communication integration. As such, the IIS may be mapped against the GERA Partial level on the Functional and Information views, at the Preliminary Design life cycle phase.
Once implemented, the IIS would become an enterprise module[224], usable in the implementation of enterprise operational systems (as shown in Fig. 3.1).
A generalised and updated version of the CIMOSA IIS is used in Annex A of ISO/IS 15704 (ISO/TC184/SC5/WG1 , 2000a) under the designation Enterprise Model Execution and Integration Services (EMEIS)[225].

3.7.4 A Sample Zachman Framework Partial Model

Zachman does not explicitly[226] describe reference models. However, as it has been shown in Fig. 3.25, it is possible to define Zachman frameworks at the partial level. A partial level framework may subsequently be either further specialised (in effect producing another, more specific reference model) or instantiated into a particular framework, applicable to the particular modelled entity.
On the other hand, various applications of the Zachman framework to specific industry types have been produced by third parties[227]. They may be considered partial models at various levels of specialisation (although they are not specifically included in the Zachman architecture framework).
The following example originates from a high-level 'business process-driven OO development' example presented in (Popkin Software , 2001). It is the model of a typical small-scale Object-Oriented software development project covering the first phase of a change process in an enterprise, i.e. modelling of the existing processes in order to better understand (and subsequently improve) the business operations.
In the following example, it is assumed that the need for the new artefact (in this case a software system) has already been identified and the decision has been taken to respond to the need by building it.
The project does not attempt to improve or re-engineer-, but rather just document the existing processes and produce a software system to reflect them. It is assumed that the business already has one or more databases containing the information, which will be used by the new application.

---

[224] belonging to the CIMOSA Enterprise Operating Environment (EOE).
[225] In ISO/IS 15704, EMEIS has the task to "provide an example of infrastructure linking the enterprise model development with the real world" (ISO/TC184/SC5/WG1 , 2000a).
[226] in the GERA sense - i.e. an explicit component in the architecture framework.
[227] similar to modelling methodologies, most third-party (and Zachman proprietary) partial solutions are not publicly available and therefore not included in this review. The reader is directed towards the web sites of such developers, previously listed in Section 3.6.4.
[228] cell content shown in italics is optional.

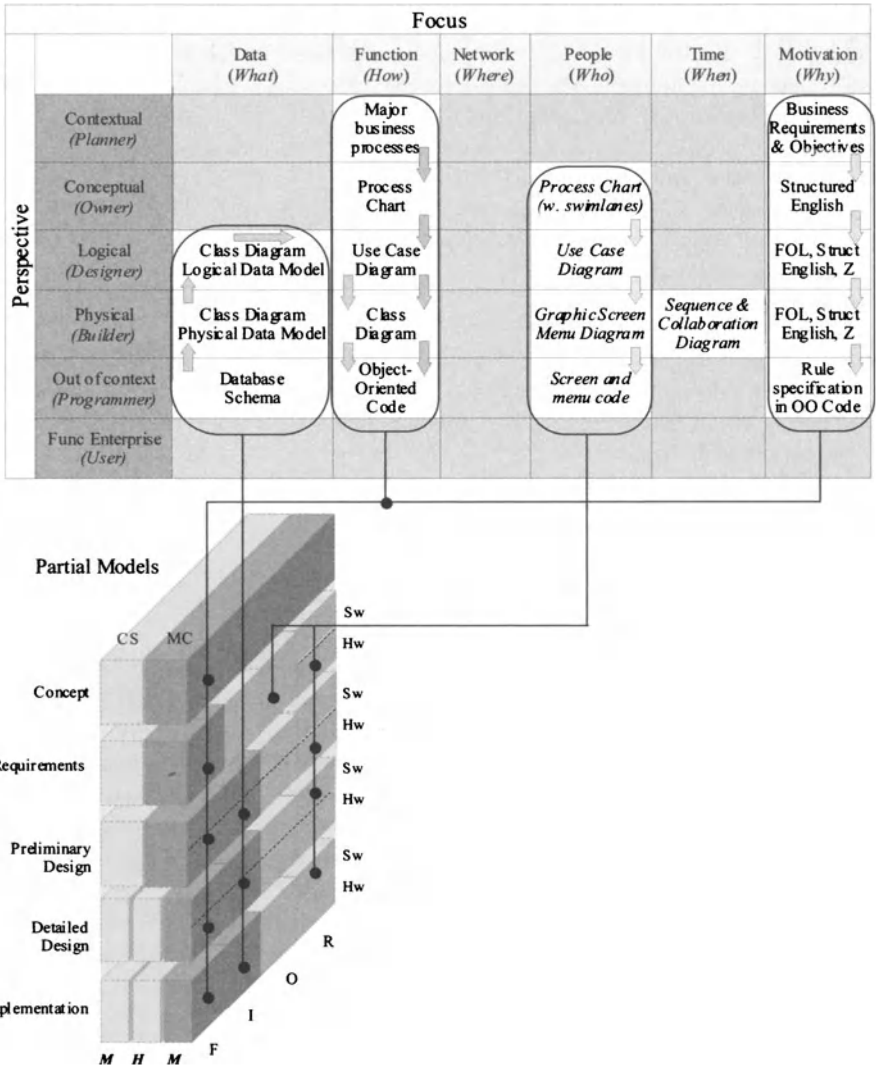


Fig. 3.51. Zachman partial model example (based on (Popkin Software , 2001))[228]and its mapping onto GERA's Partial Model level

As one can see from Fig. 3.51, the Where and When columns have not been modelled at all, the locations being considered fixed and known and the temporal aspect not complex enough to justify modelling. The Who abstraction, although also known, has been modelled from the application's point of view (in order to produce the user interface specifications and its code). The What abstraction does not include the Contextual and Conceptual perspectives since they are beyond the extent of the proposed modelling task. The 'What' (data) Logical and Physical perspectives have to be 'reverse engineered', based on the existing database system[229].
Notes about Fig. 3.51:

- the notations used for the Zachman perspectives vary widely according to the user's domain. This is reflected in the notations used in this Chapter for the Zachman framework. The equivalence of the perspectives' designations is provided below:
  - Contextual: Objectives, Scope
  - Conceptual: Business, Enterprise
  - Logical: Architect, System
  - Physical: Builder, Technology
  - Out of Context: Subcontractor, Components

Modelling the 'How' (functional) abstraction is identified as the backbone of the whole modelling effort - therefore all perspectives (i.e. GERA life cycle phases) are covered. As shown in Section 3.4.4, Fig. 3.22, the Why abstraction belongs to the functional aspect (rule modelling) therefore it is also fully covered. **NB** in the original (Popkin Software , 2001) example, the Why abstraction was modelled by means of Requirements / Test plans, due to the specific approach taken by the Popkin software tool (which partially meets the original intention of the Zachman framework).
The model presented belongs to a partial level (using levels of genericity as described in Fig. 3.25) since it is a reusable solution to a *type* of problem. It can be mapped onto the Partial level of GERA, covering the GERA life cycle phases from Concept to Detailed Design on the Management and Control side, excluding humans[230] (refer Fig. 3.51).
The GERA Function view is represented in all life cycle phases relevant to this example. The GERA Information view is represented in its Preliminary Design, Detailed Design and Implementation life cycle phases. Zachman framework's People column may map onto the GERA Organisation and Resource views as shown in Fig. 3.22. In this case however, the Organisation view is only represented at the GERA Requirements phase (using a process

[229] the existing database (i.e. the logical model and schema) may subsequently be altered as a result of the modelling process.
[230] the scope of the envisaged (software) product for this example has been established (in the Zachman Business- and/or Architect's perspectives) not to include human resource management.

chart with roles and swimlanes for the matching Zachman cell content). The GERA Resource view is covered via hardware / people aspects in the GERA Requirements- and Architectural Design phases (called in Zachman Conceptual and Logical respectively), and via Software aspects (e.g. diagrams and code) at the GERA Detailed Design- and Implementation phases (called in Zachman Physical and Out of Context).

3.7.5 C4ISR Partial Models

C4ISR includes a set of constructs called Universal Reference Resources, comprising sets of "concepts, entities, interfaces, and diagrams" (according to C4ISR Architectures Working Group (1997)). These constructs play the role of reusable, partially specialised models and consequently they may be mapped against the Partial Model level of GERA.
Figure 3.52 presents the C4ISR and GERA viewpoints in respect of C4ISR's Universal Reference Resources.

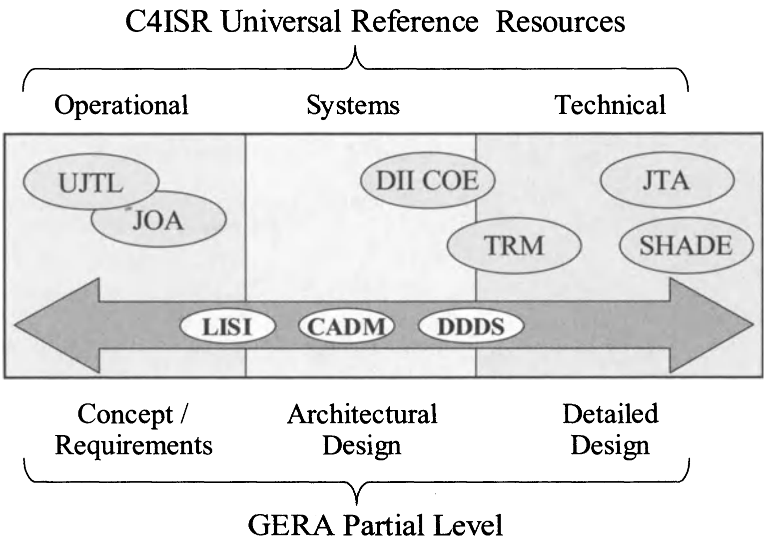


Fig. 3.52. Several C4ISR Partial Models (based on the work of C4ISR Architectures Working Group (1997))[231]

It has been previously shown that some constructs may play multiple roles within their parent architecture framework. C4ISR is no exception. For example, the Core Architecture Data Model (CADM) may be considered a reference (partial) model while at the same time representing the C4ISR metamodel.

[231] abbreviations explained in the main text

The applicability of the CADM (refer Section 3.8.3.5), Levels of Information Systems Interoperability (LISI, refer Fig. 3.54) and Defence Data Dictionary System (DDDS) spans across all C4ISR views (or GERA life cycle phases) - since their main purpose is to ensure the consistency and interoperability / integratability of the C4ISR products. Figure 3.53 presents the mapping of the C4ISR partial models onto the GERA Partial Model level.

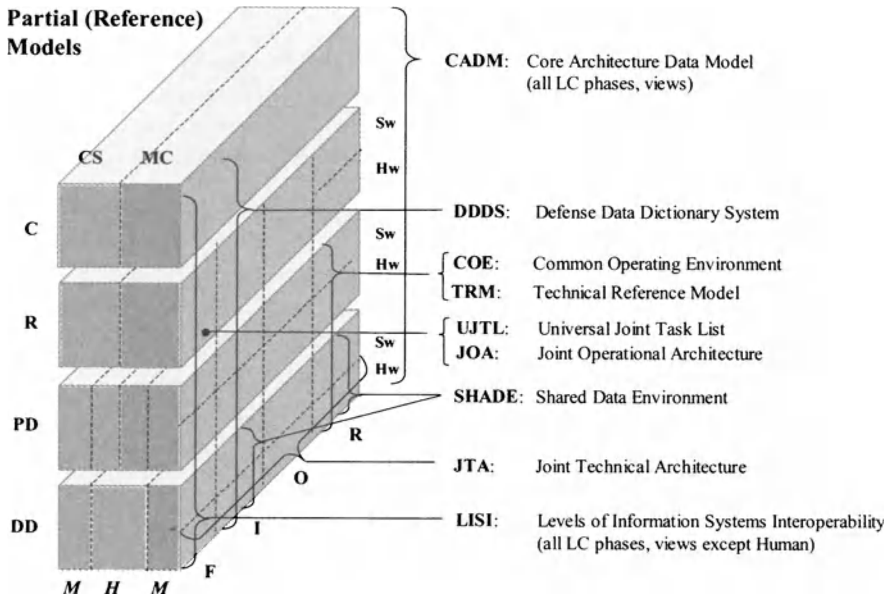


Fig. 3.53. C4ISR modelling framework mapping to GERA Partial Models level[232]

Notes about Fig. 3.53:

- CADM maps onto all (relevant to C4ISR) GERA life cycle phases and views (refer Section 3.8.3.5);
- LISI maps onto all (relevant to C4ISR) GERA life cycle phases and views except the Human aspect in the GERA Preliminary Design and Detailed Design life cycle phases.

The CADM plays the role of both a partial model and a (customisable) meta-model for the C4ISR architecture framework. For more details refer Section 3.8.3.5.

[232] this mapping covers both the Management / Control (MC) and Customer Service (CS) views of GERA. Refer Fig. 3.52 for a graphical representation of the Universal Reference Resources (URR) coverage of the C4ISR.

The Common Operating Environment (COE) describes system requirements for a resource providing infrastructure services and common support applications. As such, COE may be mapped against the Resource view of the GERA Preliminary- and Detailed Design life cycle phases[233]. COE may be considered the peer of the CIMOSA IIS[234] or the ISO/IS 15704's EMEIS.

The Technical Reference Model (TRM) provides a high-level partial model for services and interfaces in order to promote commonality and interoperability across the products developed in the US Department of Defence (DoD). TRM may be used to specify application (software) and platform (software and hardware) entities and the necessary interfaces to the external environment. As such, TRM is in relation with the Standards Technology Forecast (TV-2). TRM addresses the Systems and Technical views in C4ISR and as such maps onto the Resource view of the GERA Preliminary and Detailed Design life cycle phases (since it describes sample hardware and software resources). Currently, the TRM does not explicitly address human-specific issues.

The Shared Data Environment (SHADE) is an extension of the COE in the C4ISR Technical view, with emphasis on strategies and mechanisms for *data* sharing. Hence, it maps onto the Resource and Information views of the GERA Detailed Design level.

NB As previously mentioned, sharing data does not necessarily imply sharing a *common interpretation* of its meaning (which is information). The Integrated Dictionary (glossary) plays an essential role in information sharing, supporting common data interpretation across the architectural products.

The Universal Joint Task List (UJTL) represents a repository of tasks out of which activities may be selected and further specialised in order to define requirements for combat activities. UJTL is one of the partial models attempting to address interoperability as a key enabler to joint operations. As such, UJTL maps onto the GERA Functional aspect of the Requirements life cycle phase. The Joint Operational Architecture (JOA) aims to produce 'views of functions' (C4ISR Architectures Working Group , 1997) leading to activity models and information exchange and capabilities requirements matrices. Therefore, JOA maps onto the same GERA area as UJTL.

An essential aim of C4ISR is to provide the necessary constructs for joint operations (see UJTL above). C4ISR provides a metamodel (the CADM, described in Section 3.8.3.5) but does not (as yet) provide a detailed modelling methodology. For this reason, maintaining consistency and integratability across various architectural products is a central issue in C4ISR. Joint operations, consistency and integratability are also supported by the introduction of the concept of *interoperability.* C4ISR acknowledges the transition from user- to system interoperability requirements and their subsequent translation into system ca-

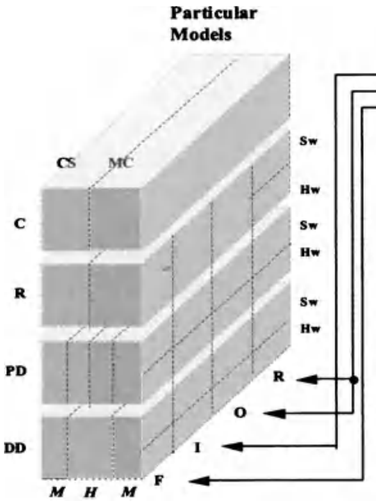
233 refer Fig. 3.6 for the basic mapping of C4ISR views onto the GERA life cycle phases.

234 although the CIMOSA IIS may exist at a somewhat more specialised (CIMOSA-linked) level.

pabilities. The C4ISR Levels of Information Systems Interoperability (LISI) construct provides both a reference model for interoperability and suitable constructs for capabilities - therefore being usable for all C4ISR views. Hence, LISI maps onto the Functional view of all (relevant to the C4ISR mapping) life cycle views of GERA.

C4ISR takes several perspectives to interoperability, defining procedures, application, infrastructure and data aspects. These aspects may be mapped onto the GERA Function (LISI Procedures / Rules and Applications), Information (LISI Data), Resource and Organisation (LISI Infrastructure), such as shown in Fig. 3.54.

In the author's opinion, the LISI levels are conceptually similar to the Carnegie-Mellon University (CMU) Capability Maturity Levels (CMM)[235] and therefore will potentially be connected to the future Capability Maturity Profile architectural product (AV-3) proposed in C4ISR.



Nature of Operational Info Interaction	Corresponding Interoperability Level		Implications: Procedures	Applications	Infrastruct	Data
Cross-domain interactive manipulation	Enterprise	4	Enterprise Level	Interactive	Multiple topologies	Enterprise Model
Shared Apps and Databases	Domain	3	Domain Level	Groupware	World-wide networks	Domain Model
Complex Media Exchange	Functional	2	Program Level	Desktop Automation	Local Networks	Program Model
Simple Electronic Exchange	Connected	1	Local / Site Level	Std. System Drivers	Simple Connection	Local
Manual Gateway	Isolated	0	Access Control	N / A	Independent	Private

Fig. 3.54. The LISI PAID views and their mapping to the F, I, R, O views of the GERA Partial Models

The Joint Technical Architecture (JTA) provides a partial model for building technical architectures (such as the technical architecture profile TV-1). As such, it refers to the Technical view of C4ISR and should encompass all GERA aspects of the Detailed Design life cycle phase. This coverage is represented in Fig. 3.53.

The Defence Data Dictionary System (DDDS) is the component of a larger partial model for a common data repository - the Defence Data Repository

[235] LISI exists on a more specialised level than CMM since it deals with a particular type of capability, which is interoperability.

Suite (DDRS). As shown in the C4ISR specification , DDRS covers "functional, technical, operational and personnel requirements received from all functional areas" (C4ISR Architectures Working Group , 1997). Thus, DDRS covers all C4ISR views and may be mapped against the GERA Information view of all life cycle phases (relevant to C4ISR). The DDDS (as a currently implemented component of DDRS) may be used as a partial model for an integrated *data* dictionary[236], hence mapping onto the same GERA areas as the DDRS (refer Fig. 3.53).

3.7.6 ARIS-compliant Partial Models

The ARIS approach employs reference models to capture process expertise and thus provide an initial partial process engineering solution to the users. In the ARIS sense, reference models fall in two main categories: procedural models and business models. The former category provides assistance with software development, while the latter category deals with activity types such as order processing.

Reference models following the ARIS concept have been developed both in-house[237] and by third-party developers. In-house reference models are available for both generic project- and industry-specific processes and are partly available in implemented form[238] as plug-in modules for the ARIS Toolset (a proprietary modelling tool, refer Section 3.8.4.1 for details). Third-party software vendors / consulting companies such as SAP or KPMG Consulting have also developed ARIS-compliant reference models for specific projects and/or industry sectors (e.g. the SAP R/3 or KPMG insurance reference models (Scheer , 1999)).

In an enterprise, there is always the potential for a gap developing between the actual business goals[239] and the business' IS (and its IT component) *perception* of those goals. ARIS aims to bridge this gap, situating its own life cycle between the Operational Business Problem (equivalent to GERA Concept) and Information Technology (equivalent to GERA Implementation, refer Fig. 3.9). Scheer (1994a) identifies the challenge of translating the business knowledge into IS-specific structures that can be further processed by the IT component. In order to support this translation task, a set of partial models (structured in accordance with the ARIS House framework) is provided in (Scheer , 1994a). This set of models is structured in accordance with the scope of the CIM Reference model (also called the Y-CIM model) presented in (Scheer , 1994b).

[236] for example, it may be used to build the Integrated Dictionary (AV-2), however covering only the data aspect (Information view in GERA).

[237] the term 'in-house' used hereafter to denote the framework developers, as opposed to external / third-party.

[238] this form satisfies the GERAM criterion of Enterprise Module.

[239] i.e. the business goals as understood by the management and technical personnel (engineers, technicians, operators, etc)

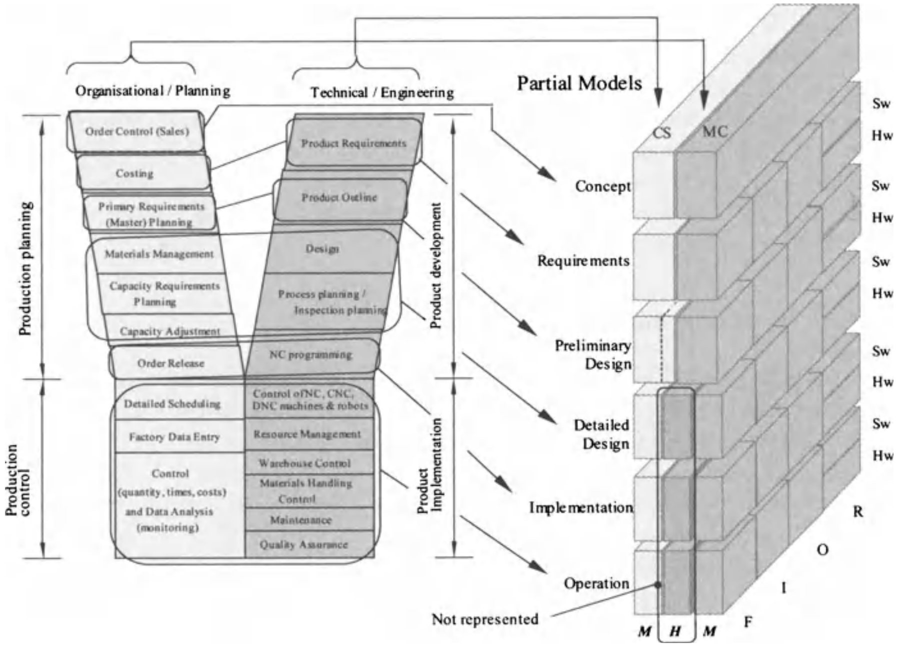


Fig. 3.55. A partial model for CIM (based on (Scheer , 1994a), p.83) and its mapping onto the GERA.

Therefore, it has been assumed that a mapping of the Y-CIM model to GERA should cover the entire set of partial models in question.
In the mapping of the Y-CIM model to GERA, it has been considered that the customer order refers to a new product, which needs to be developed (e.g. an enterprise) - and therefore phases such as Product outline, costing, design are necessary. As can be seen from Fig. 3.55, the GERA Identification life cycle phase is not explicitly covered - it is assumed that the customer has identified he need for the product in question. In addition, the Obsoletion GERA phase is not covered in the CIM reference model. Order processing / control has been mapped onto the Concept phase of GERA since it represents the commitment to take action in order to respond to the identified need for a product (or its change). Typically for a CIM model, the Operation phase of GERA is modelled in detail by the Y-CIM model by means of several phases. The human aspects required by GERA are not covered in this model.
For further information regarding ARIS partial models refer (Scheer , 1994a).

3.7.7 Conclusion About Partial Models

Reference architectures attempt to model the structure and life cycle of complex artefacts. In their bid to tackle this complexity, architectures often em-

ploy a 'divide and conquer' approach, resulting in more than one model (each reflecting a particular *view* of the artefact modelled). Therefore, GERAM compliant[240] reference architectures must provide constructs and frameworks expressive enough to construct and hold these models - which implies a certain degree of complexity. This complexity makes reference architectures less attractive to business users, since they face the challenge of learning several[241] architectures (each with a framework, modelling methodologies, proprietary languages, etc).
Partial models play an important role in alleviating these difficulties, since they are essentially templates, which may potentially accelerate the learning process (learn by example) and save resources by means of reuse. Therefore (and according to GERA), the more partial models a reference architecture encompasses, the easier to learn and use (and therefore the more attractive) it is to the users. Intuitively, building a partial model is a two way process: on one hand they are produced by specialising[242] generic models, but on the other hand they need to be validated via several good[243] particular models obtained by instantiating[244] the partial model. In addition, the validation process is almost certain to modify and enrich the partial model.

3.8 Other Relevant Constructs

This section attempts to cover the rest of the GERAM components (shown lightly shaded in Fig. 3.1) and other relevant components of the GERA, such as enterprise entity types and their recursivity. Very often, the concepts described in this section are only present in an implicit form in the reviewed architecture frameworks. If the concepts cannot be identified at all, a short

[240] Refer Chapter 2 and (ISO/TC184/SC5/WG1 , 2000a) for GERAM compliance criteria.

[241] none of the existing architectures reviewed covers by itself all of the aspects specified in GERA in the necessary level of detail.

[242] specialisation seen as restricting the range of (or making constant) one or more variables in a generic or partial model. In this context, 'partial instantiation' does not seem to be a suitable substitute for 'specialisation'. In the GERA framework, the more specialised a partial model is, the more towards the particular model level (the right-hand side on the 'Instantiation' axis) it is situated. For a concrete example of partial model specialisation applied to systems' life cycle standards, refer (Noran , 2000a).

[243] that is, complete and expressive for the purpose they have been built for and understandable to the intended audiences (Bernus et al. , 1996a). For enterprise models evaluation methods refer (Fox and Gruninger , 1998).

[244] instantiation seen here as making constant *all* the variables. A significant exception however constitute models containing run-time variables (i.e. meant to be set and re-set during operation). Such models would in essence constitute *partial executable* models, producing a new particular model at run-time with each new reconfiguration.

what-if analysis is performed to determine (at a high-level) whether the respective framework is compatible with the concept in question - and therefore *could* potentially accommodate it.

3.8.1 Enterprise Entity Types

The identification of the target Enterprise Entity Types is an essential activity in enterprise architecture. The concept of entity type provides the means for entity classification and hence allows a more structured approach towards enterprise integration. Bernus and Nemes (1994) initially defined four types of enterprises entities:

- Strategic management;
- Engineering;
- Manufacturing;
- Product.

GERA has added a fifth type of entity, the Methodology. For details on entity types refer (ISO/TC184/SC5/WG1 , 2000a) and Chapter 2 of this book.

3.8.1.1 Entity Types in PERA

PERA includes an entity type construct without explicitly calling it a 'type'. It simply identifies various enterprise *kinds* and attaches numbers to them[245] similar to the approach used by GERA. Therefore PERA complies with the GERA concept of entity type.

3.8.1.2 Entity Types in CIMOSA

CIMOSA does not explicitly define enterprise entity types, although its framework is compatible with this concept (refer Section 3.8.2.2, Fig. 3.57 for an illustration of this compatibility). Several frameworks may be used to define enterprise entity types, similar to the GERA approach[246].

3.8.1.3 Entity Types in GRAI-GIM

GRAI-GIM does not explicitly define enterprise entity types. However, there are no major obstacles in using the GRAI modelling framework (shown previously in Fig. 3.19) or the GIM Structured Approach (refer Fig. 3.3) for representing various types of enterprises and analysing their interaction, in a similar way that GERA modelling framework is being used in Fig A.5 of ISO/TC184/SC5/WG1 (2000a).

[245] e.g. 'enterprise #2' is an engineering entity, while 'enterprise #3' is a manufacturing entity.

[246] this similarity originates from CIMOSA being a major contributor towards the GERA modelling framework.

3.8.1.4 Entity Types in Zachman

Zachman does not provide a specialised enterprise entity type concept. That said, Sowa and Zachman (1992) do identify separate frameworks describing various kinds of entities, such as product, manufacturer[247], information system or CASE tool manufacturer (refer Fig. 3.58). This denotes that the Zachman architecture framework *does* recognize entity types in the GERA sense. Also in Zachman (like in GERA), not all of the entity types are necessarily *enterprise* entities[248]).

3.8.1.5 Entity Types in C4ISR

C4ISR does not explicitly identify enterprise entity types in its framework. Enterprise entity types may however be identified within the Department of Defence (DoD), such as:

- Strategic Management Entities (e.g. HQ ADF[249], HQ Army, HQ Navy, HQ AF[250]);
- Warfighting Units (Forces), Joint Operations, Command and Control Entities (e.g. Australian Theatre);
- Capability development and support entities (e.g. DMO, DISG, Surveillance, Intelligence);
- Capability Development Projects;
- Corporate-wide Service entities.

This suggests that the C4ISR framework is compatible with the enterprise entity type concept. A formal recognition of this concept though would focus the enterprise modelling effort on specific types and therefore make the C4ISR architecture deliverables more specific.

3.8.1.6 Entity Types in ARIS

ARIS does not explicitly define enterprise entity types in the GERA sense. However, according to the structuring of the entire ARIS concept, an enterprise entity type can be explicitly created and stored at the meta-meta level (refer Fig. 3.30) as a specialisation of the Object Type construct. The enterprise entity type construct would then populate the ARIS meta level and could be specialised into a discrete range of values, such as e.g. {enterprise engineering, enterprise, product, ...}, at the application (model) level.

[247] which is merely called 'enterprise' in (Sowa and Zachman , 1992).

[248] e.g., GERA defines a methodology entity (type 5) which is not an actual enterprise. Similarly, Zachman defines a 'knowledge management' framework which may not necessarily be an actual enterprise.

[249] Headquarters Australian / American etc. Defence Force

[250] Headquarters Air Force

3.8.2 Recursivity of Entity Types

The recursivity of entity types as defined in (Bernus and Nemes , 1994) and ISO/TC184/SC5/WG1 (2000a), acknowledges the existence of relations of a recursive[251] nature between life cycles of various enterprise entity types. It is however emphasized that, according to GERA, only the Operation life cycle phase of an entity may influence other entities' life cycle phases (refer ISO/TC184/SC5/WG1 (2000a) for details). This section aims to acknowledge attempts of the reviewed architecture frameworks to define relationships of a recursive nature between (types of) enterprises[252]. If the framework in question does not define such recursivity, it is briefly analysed to determine if it is at all *compatible* with the concept of entity recursivity.
NB In the following, recursiveness is understood as repetition[253] and refers by default to an active and direct influence of one entity in the development of another entity. An alternative to this direct influence is the use of one or more deliverables obtained during the creation of an entity into the development of another, by a third, *operating* entity. This alternative meaning is used (and tested for) in case the framework in question does not explicitly support entity recursivity.

3.8.2.1 8.2.1 Entity Type Recursivity in PERA

PERA has defined the concept of interaction of enterprise entities in its subsequent extensions (Li and Williams , 1994). While agreeing to the number of maximum four possible enterprise types (as originally stated in the work of Bernus and Nemes (1994)), it claims that the 'chains' formed using the four entity types may be of infinite length.
As can be seen from the example in Fig. 3.56, PERA appears to contradict the GERAM recursion concept twice by allowing an enterprise entity kind to support another within a life cycle phase other than Manifestation (GERA Operation). This is the case of the Manufacturing Entity, apparently providing support for the Design entity and the Construction entity before it even started operating. However, as can be seen from the left hand side of the figure, the Detailed Design content supporting the Construction entity is provided by another, *operating* Engineering (Design) entity. Thus in fact there is no contradiction, since an entity A may contribute to the development of entity B, and in turn entity B can contribute to the development of a *future state* of entity A.

[251] recursivity example: an enterprise of a particular type X may support another enterprise of the same type X - therefore the 'support for enterprise type X' function may be defined in terms of itself (and possibly, an additional invariant used to stop the recursion).

[252] and implicitly between the modelling frameworks relating to the enterprises in question

[253] an alternative meaning could be e.g. decomposition.

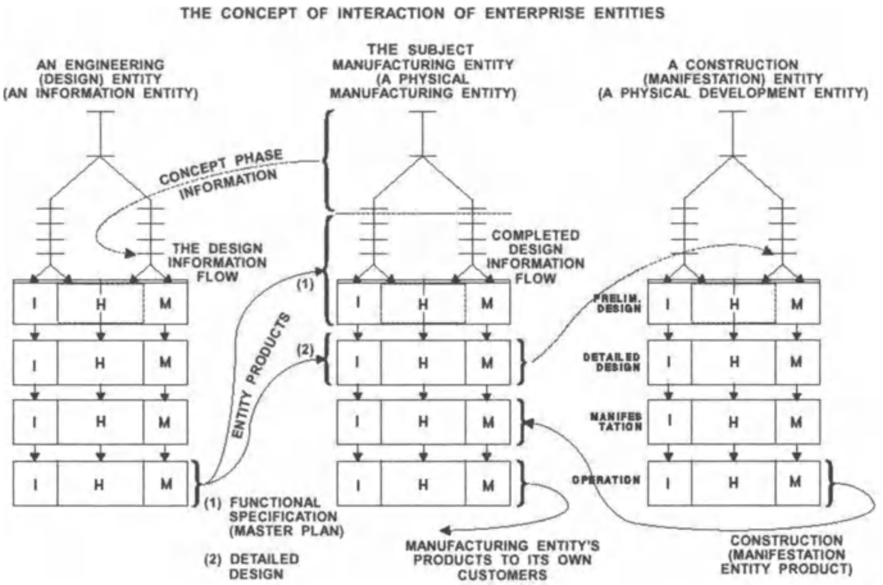


Fig. 3.56. Interaction of Enterprise Entities (based on (Li and Williams , 1994)).

The other apparent contradiction is the 'concept phase information' provided by the Subject entity (not yet even designed) to the Design entity. In reality, the Concept phase information is also produced by another operating entity. This entity may be a Strategic Management entity (as defined in GERAM) or the management of a Manufacturing enterprise type (or even of the enterprise in question in the case of re-engineering)[254].

3.8.2.2 CIMOSA and Recursion

CIMOSA does not explicitly define entity recursion although its modelling framework is compatible with the concept of entity type recursion as defined in ISO/TC184/SC5/WG1 (2000a). This is mainly due to the similar structures of the CIMOSA and GERA modelling frameworks (refer Fig. 3.57). There are however some aspects to recursion, which are specific to the CIMOSA modelling, approach.
In CIMOSA, an enterprise is seen as a network of interacting, CIMOSA- and non-CIMOSA-compliant domains. Therefore, an enterprise could be influenced by other enterprises via their domain processes' events and/or results.

[254] in which case an arrow would be needed, pointing from the Manufacturing entity's Operation phase to its own Concept phase.

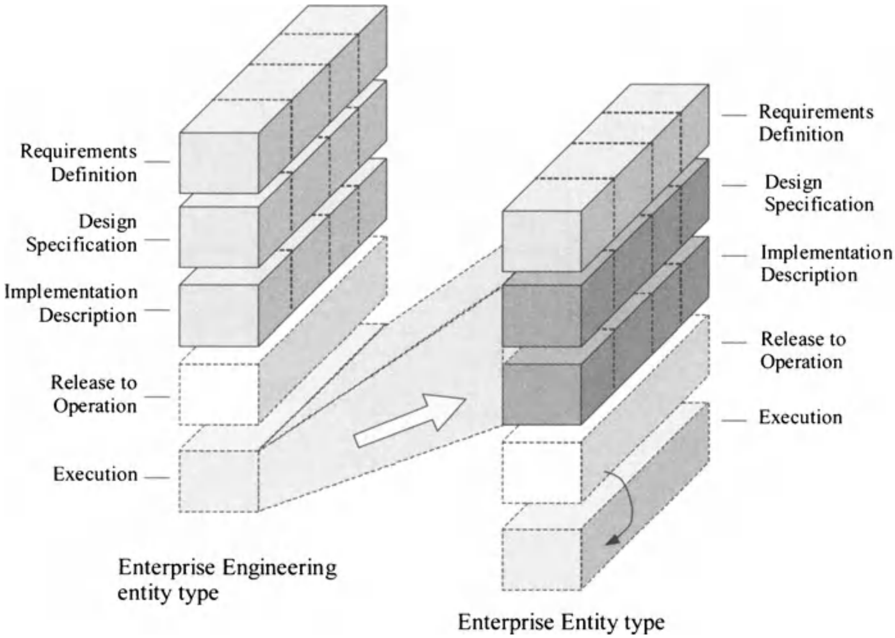


Fig. 3.57. GERA recursion concept applied to CIMOSA (based on (ISO/TC184/SC5/WG1 , 2000a))

This influence may occur either in the initial entity modelling-, or in the subsequent maintenance phases[255] (refer Fig. 3.4).
Either way, an enterprise must be operating (and controlled via its executable model in the user-defined Enterprise Operational Environment, according to CIMOSA) in order to be able to influence the development or maintenance processes of other enterprises[256].

3.8.2.3 Recursion in GRAI-GIM

As shown in Fig. 3.42, GRAI in its Structured Approach does not specifically cover the Operation phase - therefore it cannot explicitly express entity recursiveness in the GERAM sense.

[255] comprising one or more of the other life cycle phases towards the purpose of model update, modification and/or extension (Vlietstra , 1996).
[256] As shown in the following Sections, phases from the life cycle of non-operating entity *may* be used to influence the life history of another entity. This however must be accomplished by a third, operating entity (which in fact confirms the GERA meaning).

That being said, it *is* possible for GRAI deliverables produced in various life cycle phases (e.g. Analysis, User-oriented design, etc) of an entity 'A' to be included in the life cycle phases of another entity, 'B'. This task however, has to be accomplished by an third, built and *operating* entity 'C', which models the entity 'B' using e.g. the GRAI Structured Approach. In conclusion, the GRAI architecture framework is compatible with- and indirectly supports the concept of entity recursion.

3.8.2.4 The Recursion Concept in the Zachman Framework

Zachman identifies three dimensions to the recursivity:

- relation between frameworks associated to entity types;
- applying the logic of the Zachman framework to the framework itself (applicable, but in the author's opinion relevant only for the first recursion step, mainly due to the exponentially increasing complexity)[257];
- versioning of the Zachman frameworks. As previously shown, since it is reflecting the AS-IS, TO-BE and possible intermediate states, versioning is also a life history concept. Versioning based on recursion may imply that a framework at a given point in time is expressed in terms of all of the previous versions of that particular framework. This may not always be true (it is often the case that a current version of a particular product is obtained from the most recent version only (i.e. only the first recursion step)).

Out of these three aspects of recursivity, only the first type applies to entity types via their modelling frameworks, and hence matches the scope of this Section. The second and third recursivity aspects are common to all of the reviewed frameworks, even if it is not always explicitly stated.
It has been previously shown that the GERA concept of enterprise type recursion assumes that it is always the Operational phase of an entity life cycle that may influence another entity ISO/TC184/SC5/WG1 (2000a). The recursive relation in GERA explicitly refers to 'defining, creating, developing and building' the influenced entity.
The Zachman approach to entity type recursivity is somewhat different. The relation between entities is restricted primarily to a *descriptive* type. Figure 3.58 represents this descriptive relation between frameworks - identifying products, enterprises[258], information systems and CASE tool manufacturers. The descriptive character of the relation is revealed in (Sowa and Zachman ,

[257] **NB** this type of recursion applies to any architecture framework (including GERAM), because if an architecture deliverable needs to be developed the life-cycle of this deliverable should (and can) be described by the framework itself – often considered as a 'sub-project', such as customary in systems engineering;
[258] or manufacturing entities, in GERA terms.

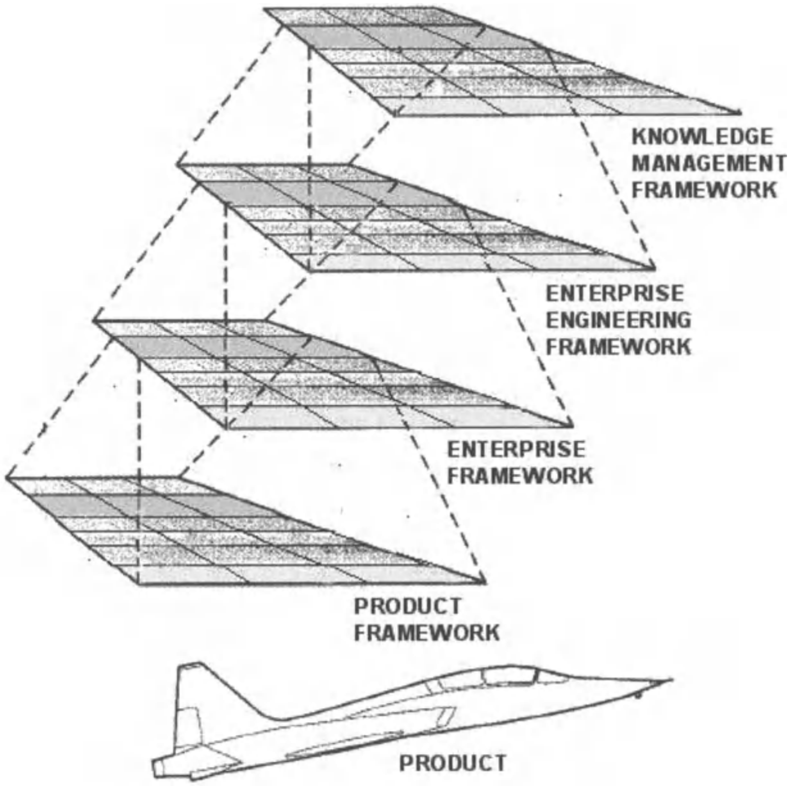


Fig. 3.58. Framework recursivity in the Zachman framework (based on (Sowa and Zachman , 1992) and (Inmon et al. , 1997))

1992), which argues that *row number 2* of the manufacturing entity's framework (containing the owner / business view of that entity) contains in fact the complete description of the manufacturing entity's product(s)[259]. Hence, the manufacturing entity framework's row number two must in fact be a *meta-model* of the product framework.

While the concept shown in Fig. 3.58 is valid from a theoretical point of view, it may not be able to cover some situations such as dynamic self-reconfiguration of a manufacturing entity at run-time (row six)[260]. In GERA terms, row two

[259] where the first cell from the left (Owner's View intersected with the Who column) contains all data descriptions relating to the product(s), second cell (Owner's View intersected with the How column) contains all the business processes necessary to manufacture the product(s), and so on.

[260] e.g. a robot, or entire manufacturing cell may change its configuration (physical or control) during operation.

represents the gathering of user requirements for the manufacturing entity. The manufacturing entity is not yet operating and thus it cannot actually *do* anything. In the case of re-engineering, the enterprise *is* operating in parallel with the gathering of new requirements, however any future products are just being described as *part of* the requirements gathered.
If other actions, such as develop, build etc are to be considered within the recursive relation, then it is not necessarily the case that a framework should cover *the whole* of another, but rather only the relevant parts. For example, the manufacturing entity may have its Context and Business Owner's views defined by a management entity and its Architect's and Builder views defined by an engineering implementation entity (refer ISO/TC184/SC5/WG1 (2000a) for a typical example).
Therefore, a recursive relation involving definition, development, building, etc of should originate from row six of the 'parent' (supporting / influencing) framework(*s*) to the relevant parts (not necessarily the whole) of the 'child' (supported / influenced) framework[261].
Row 2 may be considered in defining recursion only in the sense already explained e.g. in Section 3.8.2.3, i.e. if a third framework, of an operating entity, is involved.

3.8.2.5 Recursion in C4ISR

As previously shown in Section 3.8.1.5, entity types may be implicitly identified within the C4ISR framework. Recursion is however not explicitly present in C4ISR. In addition, C4ISR does not cover the Operations life cycle phase, which would allow modelling of the influences that an entity type may have on other entities.
Similar to the case of the GRAI architecture framework (refer Section 3.8.2.3) however, it is possible for C4ISR architectural products developed for the modelling of an entity to be used in supporting the modelling effort of another entity. This task however, has to be accomplished by a third, operating entity[262], which is involved in the modelling of the recipient entity.

3.8.2.6 ARIS Recursion

As shown in Section 3.8.1.6, ARIS does not explicitly support the notion of enterprise entity. Therefore, the concept of recursiveness of entity types is not directly applicable, in the GERAM sense. Similar to e.g. GRAI or CIMOSA, it is apparently possible for an entity (which finds itself in the ARIS-defined Operation- or Maintenance life cycle phase) to influence other

[261] a more extensive coverage of this aspect will be provided in a separate paper (at the time of writing, in preparation within Griffith University (Australia)).
[262] which may-, or may not be of the same C4ISR implicit type as the modelled entity.

entity / entities (which entities may only be in their Requirements Definition or Design Specification life cycle phases - refer Fig. 3.9).

3.8.3 Generic Enterprise Modelling Concepts

The Generic Enterprise Modelling Concepts (GEMCs) of a reference architecture and methodology are the glue that holds it together and ensures the consistency and compatibility of its components. ISO/TC184/SC5/WG1 (2000a) sets the requirements that should be met by generic modelling concepts and briefly describes their components: glossaries (an explanatory collection of terms used), metamodels (used to describe the meaning of the modelling constructs of the modelling languages[263]) and ontologies (most formal descriptions of the theories on which the architecture and methodology are based). This section will attempt to identify these components for each of the reviewed architectures.
The origins and principles that each reviewed architecture is based upon will also be very briefly stated, based on the publicly available information at the time of this writing.

3.8.3.1 Generic Modelling Concepts in PERA

PERA was developed in 1990s as a response to the need for a methodology for a previously developed CIM model (the Purdue Reference Model for CIM, later to evolve into a standard (Instrument Society of America (ISA , 2000)). A model of the methodology has been developed in graphical form (the PERA 'windchime'). The authors have then realised that they had actually developed a type 2 architecture[264], which has become the Purdue Reference Architecture. Formal models of PERA have never been an issue as PERA is aimed towards the non-computer science educated users.
A Glossary for the Purdue Architecture and its associated methodology is included in (Li and Williams , 1994). The glossary has initially been prepared by a subcommittee of Industry-University Consortium (1992). Subsequently the glossary has been revised and extended in order to become GERAM-compliant.
As shown in Section 3.5.1, PERA does not explicitly prescribe or recommend a complete set of proprietary or third-party languages. As such, there is no obvious need for a metamodel. **NB** the users of PERA (e.g. enterprise architects / modellers) will clearly need to go beyond the PERA level of detail and adopt languages and tools (according to the PERA recommendations), which in their turn must be based on metamodels and underlying ontologies.

[263] metamodels are often expressed as an 'information' (ER / IDEF1x) model or UML class diagram describing the modelling *language in question.*

[264] since the PERA diagram described in detail the phases in the life cycle of an enterprise integration project or deliverable. Therefore, PERA is in fact an architecture of the life cycle (or a life cycle architecture).

3.8.3.2 CIMOSA's Generic Enterprise Modelling Concepts

CIMOSA draws on information system and organisation theories, its process model being based on process algebras (Kosanke and Vernadat , 1996). CIMOSA's approach to enterprise modelling is based on the Function - Behaviour - Structure paradigm developed by the AMICE consortium, based on a Process / Activity / Operation approach (CIMOSA , 1996).
The CIMOSA Technical Baseline document contains metamodels (describing the relations between the CIMOSA modelling language constructs) for each of the CIMOSA life cycle phases. Within each life cycle phase, metamodels are provided for the CIMOSA genericity levels (generic, partial, particular) and the CIMOSA views (function including behaviour, information, resource, organisation). The metamodels are described using a notation derived from the IDEF1x language. A CIMOSA metamodel is also available in the above-mentioned document for the relation between generic, partial and particular concepts, including the meaning given by CIMOSA to specialisation, instantiation and occurrence.
A high-level CIMOSA metamodel is also available in (Vernadat , 1996). Fig. 3.59 presents this metamodel, enriched with the organisational concepts described in text form. As argued earlier in this Chapter, such a high-level metamodel is not entirely satisfactory to directly support modelling (it has a low usability for the actual modelling purpose). Its low complexity makes it however easy to understand and ideal to communicate CIMOSA first principles to an audience familiar with the language used to express the metamodel (in this case, UML class diagram). Once consensus is achieved over a unambiguous interpretation of this metamodel, more complex metamodels (such as the detailed metamodels contained in the CIMOSA baseline for genericity levels and views) may be developed.

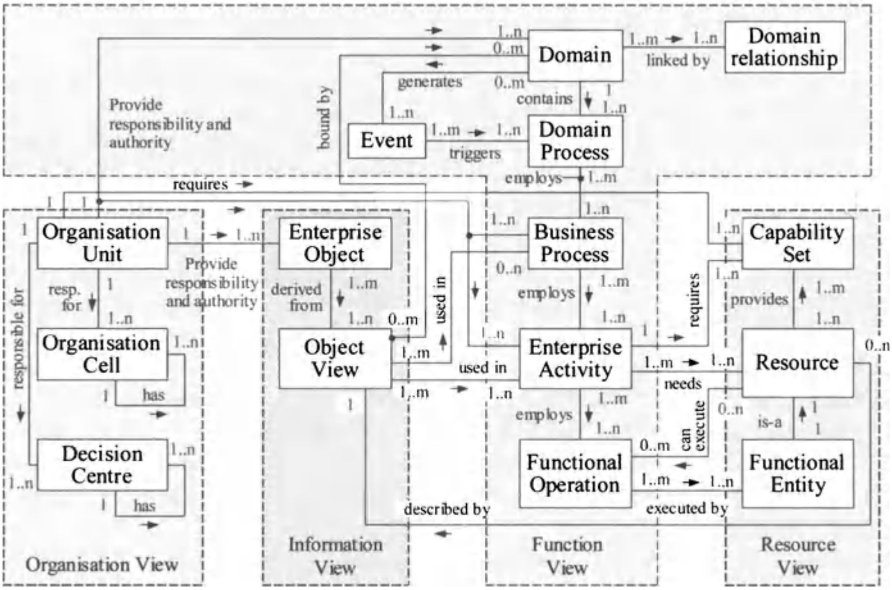


Fig. 3.59. CIMOSA metamodel (based on (Vernadat , 1996; CIMOSA , 1996; CEN/TC 310/WG1 , 2002)[265]

3.8.3.3 GRAI-GIM Generic Modelling Concepts

GRAI-GIM is based on several theories such as systems, hierarchical control, discreet activities and production management (Chen et al. , 1997).
To the author's knowledge, at the time of this writing there is no metamodel or glossary associated with the GRAI-GIM.

3.8.3.4 Generic Enterprise Modelling Concepts in the Zachman Framework

Zachman's framework was initially aimed at the enterprise's information subsystem. It is based on building and manufacturing industry approaches translated to the information systems, as described in (Zachman , 1987). John Zachman found that the manufacturing and building industries in fact asked the same questions as the information systems discipline. He then observed that while the industries organised the answers to the questions by ordered roles, information systems did not (Martin and Robertson , 2000). This has led to the idea of using the industry approach in information systems, which

[265] Objective, Integrity constraint , Declarative / Behavioural Rule, Resource Cell etc constructs not shown for clarity.

has ultimately resulted in a framework grid, organised by roles and questions (or *abstractions*).

Later on, in (Sowa and Zachman , 1992) the framework has been extended and its scope widened in order to cover the enterprise itself. In addition, the framework has been partially defined (formalised) in terms of conceptual graphs.

The Zachman framework has an underlying metamodel partially described in (Sowa and Zachman , 1992). The representation uses similar notations to the Entity Relationship modelling language, albeit less rigorous. For example, GERAM requires the metamodel of an architecture framework to show basic constraints, such as cardinalities. From this point of view, the Zachman meta-model represented in (Sowa and Zachman , 1992) falls short. Notwithstanding this, the metamodel is still a good source of information regarding the structure of several perspectives and abstractions of the Zachman framework.

For example, from the metamodel representation (partially shown in Fig. 3.60) it can be seen that the various cell contents are connected along the same row. This means that, similar to other modelling frameworks, the various abstractions (views) referring to the same perspective (life cycle phase) are complementary[266].

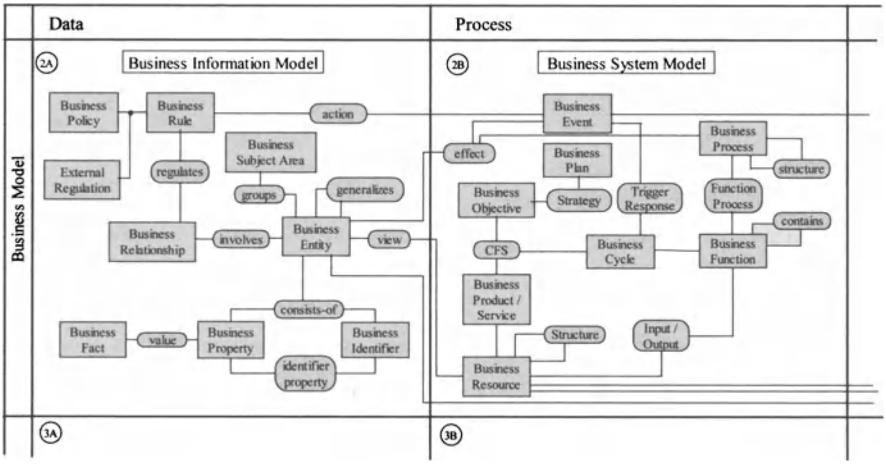


Fig. 3.60. Extract from the Zachman framework metamodel (based on (Sowa and Zachman , 1992))

266 for example the inputs / outputs of the functions represented in the How abstraction are entities that may be further described in the What abstraction.

The meta-model shown in Fig. 3.60 could become fully GERAM-compliant if extended to the whole framework and with proper cardinality constraints in place.
At the time of this writing there is no publicly available Zachman framework glossary known to the author of this Chapter.

3.8.3.5 C4ISR's Generic Enterprise Modelling Concepts

As previously shown, C4ISR's views are not directly supported by existing Systems Engineering formalisms and tools. Some of C4ISR's architecture products however are typical of the Structured Analysis and Object Oriented paradigms. There are also commonalities with the Zachman framework, e.g. the Data, Function, Network and People abstractions in Zachman may be traced in the C4ISR views. Mappings between the two, and other frameworks have been attempted[267]. It has also been suggested that C4ISR can provide templates and guidelines for modelling the enterprise features that correspond to the Zachman cells. The similarity with Zachman suggests industry influences (e.g. building and manufacturing). The conclusion is that C4ISR inherits from- and builds on each of these approaches[268]; however, a clear and unique theoretical influence may not be inferred[269].
As previously stated, the C4ISR Core Architecture Data Model (CADM) construct is in fact both a partial model and a customisable metamodel. As such, the CADM aims to describe the structure of the modelling languages used to construct some of C4ISR's architectural products (e.g. activity models etc). As shown in Section 3.5.5 however, not all modelling constructs proposed by C4ISR are described in the CADM. Figure 3.61 presents a high-level IDEF1x data model of the CADM. The full description of the CADM and an associated data dictionary are presented in a separate CADM document[270].
In order to increase the flexibility of its products, C4ISR allows controlled modifications of the metamodel. Such modifications may be useful, but are invariably fraught with the risk of inconsistencies and loss of expressiveness towards the intended audience[271]. Therefore, all metamodel modifica-

[267] e.g. with other US government architecture frameworks, such as the Federal Enterprise Framework (US Federal Govt) and Treasury's Enterprise Framework (Sowell , 2000).
[268] and implicitly it also builds on their ontologies.
[269] the C4ISR mission statement may be of further clarification: "The purpose of C4ISR architectures is to improve capabilities by enabling the quick synthesis of 'go-to-war' requirements with sound investments leading to the rapid employment of improved operational capabilities, and enabling the efficient engineering of warrior systems" (C4ISR Architectures Working Group , 1997).
[270] a full review of the CADM document is beyond the purpose of this Chapter. The reader is directed to the document web site, at the time of this writing http://www.cisa.osd.mil.
[271] changing the metamodel of a modelling language by e.g. adding new concepts without providing accompanying ontologies to define their meanings may make

tions/extensions must be fully documented and accompanied by suitable ontologies.

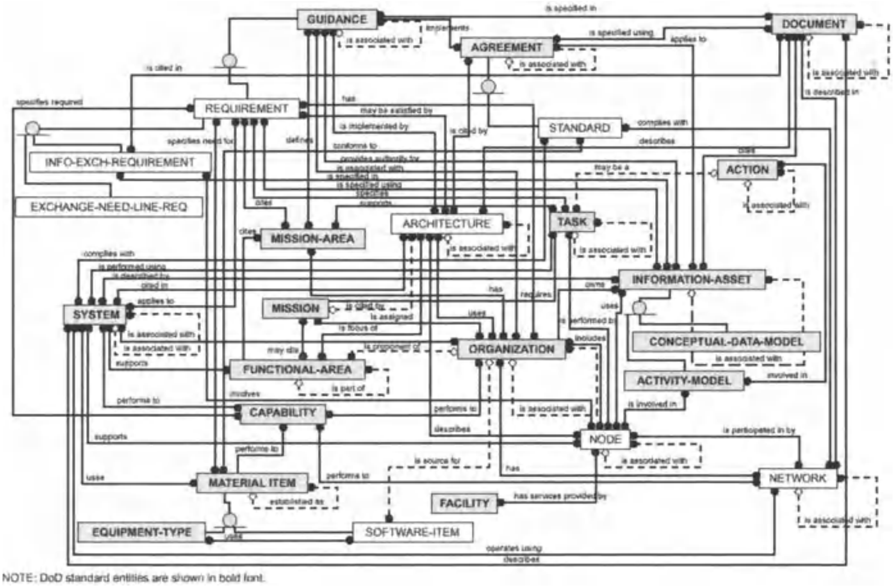


Fig. 3.61. The C4ISR metamodel (C4ISR Architectures Working Group , 1997) [272]

C4ISR also includes an Integrated Dictionary (AV-2) product which may be regarded as a glossary belonging to the Generic Enterprise Modelling Concepts, as described in GERAM (refer the GEMCs in Fig. 3.1). The integrated dictionary, together with the metamodel (the CADM) with its associated data dictionary and any available ontologies are intended to provide for the consistency, interoperability and integratability of the C4ISR products.

3.8.3.6 The ARIS Generic Enterprise Modelling Concepts

ARIS is based on the business- and systems theories as shown in (Scheer , 1999). As such, it attempts to model both the static (architectural) and dynamic (behavioural) aspect of the entity in question. ARIS intends to bridge the gap between the Business Concepts and Information Technology as previously shown in Fig. 3.9.

the altered language incomprehensible for audience trained in the use of the original language.

[272] DoD Standard Entities are shown in bold font and shaded.

As with other frameworks, the ARIS concept is defined via a set of metamodels, which populate the ARIS Meta level[273] (refer Fig. 3.30). Similar to CIMOSA, ARIS includes a set of high level metamodels and several detailed metamodels applicable to the specific ARIS views for most of the life cycles. The high-level metamodel shown in Fig. 3.62 (which only shows constructs applicable to the Requirement Definition life cycle phase) may be used for a birds-eye view of the ARIS concept. This metamodel may not be directly used to produce detailed models, but it is useful to communicate the main constructs of the ARIS concept and their relationships. For example, Fig. 3.62 confirms that the Control view does not contain any unique constructs, but rather relations between the other views (which is in line with its declared purpose).

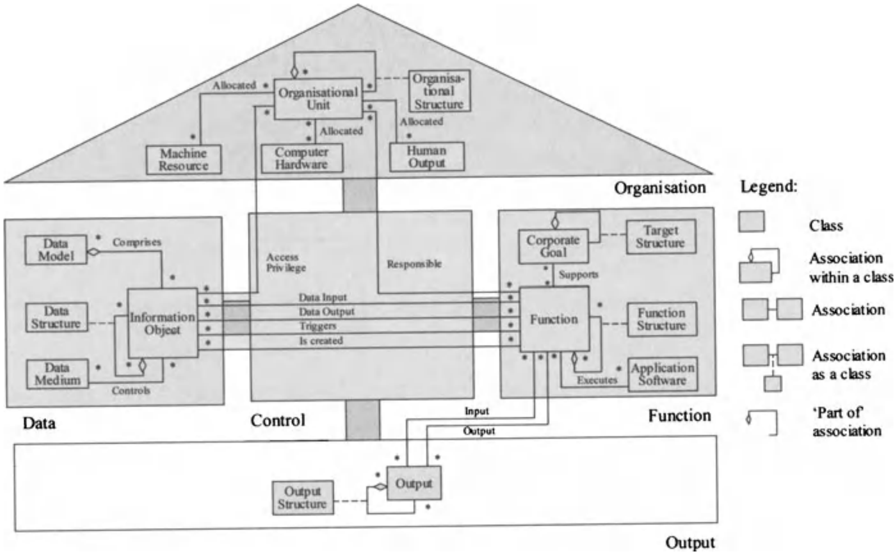


Fig. 3.62. ARIS high-level metamodel[274] (based on (Scheer , 1999))

An ARIS detailed metamodel (or information model) may also be considered to be the database schema of a repository containing ARIS models[275].

[273] the ARIS Meta level is described as containing the ARIS metamodel, the ARIS modelling framework (at the generic level), the ARIS meta-business process (similar to the graphical representation in Fig. 3.29) and the database schema for the ARIS repository.

[274] selected entities from the Requirements Definitions level shown only.

[275] such as implemented in the ARIS Toolset modelling tool.

3.8.4 Enterprise Modules

Libraries of modules are commonly used in several engineering disciplines[276] in order to build complex systems. Similar to the Object-Oriented encapsulation and information hiding principles, the users only need to know the public variables / interfaces (specifications) of such a module in order to be able to use it[277]. In enterprise integration terms, the business management would be able to plug together trusted process components in order to create a business (Bernus , 1999). Enterprise modules in the GERA sense represent instantiated partial models, where e.g. all build-time variables are fixed (in value or range) and all run-time variables are publicly known. GERA makes a special mention of the Integrating Infrastructures as a particularly important (and therefore essential within an architecture framework) enterprise module[278].

The reader is also reminded that, according to the high-level GERAM meta-model presented in Fig. 3.1, modelling tools which are 'compliant' with a specific architecture (such as those described in Section 3.9.2) may also be considered enterprise modules, which implement languages and methodologies defined by the architecture framework in question.

3.8.4.1 Enterprise Modules in the Reviewed Architectures

Few of the reviewed architectures explicitly describe Enterprise Modules. However, one must remember that enterprise modules as implemented partial models are subject to the currently available- and ever evolving information and manufacturing technologies. Therefore, most architecture frameworks limit their coverage of enterprise modules to sets of descriptions and/or requirements, which are covered in various degrees of detail in reference models.

For example, when using PERA the modeller may decide to select suitable modules so as to implement the deliverables specified in the PERA diagram, at each level. PERA only makes high-level recommendations in this regard, however other PERA guidance may be available[279].

As shown in Section 3.7.3 and Fig. 3.50, CIMOSA IIS provides services for model engineering and operation. As such, the implemented CIMOSA IIS

---

276 a classic example is electronic engineering, where a module may be e.g. an integrated circuit comprising several discrete electronic components (transistors, resistors, etc).

277 there may be however cases where the internal structure of the module has to be known - either for trust building (by visibility) or customisation / optimisation of a specific module (which may need reverse-engineering of an otherwise 'opaque' module).

278 according to GERA, this type of module may be obtained by specialising and instantiating the integrating services technology partial model.

279 e.g. 'examples' or partial models (such as e.g. the Rathwell and Williams (1996) example), containing enterprise modules specifications which may be employed to assist in the process of module selection

represents a CIMOSA module which may be used in constructing a complete Enterprise Operating System (refer Fig. 3.1).
Another example of proprietary modules is the GIM technological components library ((Chen et al. , 1997), in development), which GRAI-GIM will use to provide both Information Technology- and Manufacturing Technology modules.
To the knowledge of the author, there are no in-house Zachman's Framework enterprise modules published at the time of this writing. One may however consider third-party software packages which partially- or fully support the Zachman framework[280] as being specialised Enterprise Modules.
C4ISR uses *nodes* to describe enterprise modules. However the human modules are not properly covered - the node construct for people does not have suitable attributes such as skills / competencies. The third-party software modules available for C4ISR[281] may also be added to the C4ISR module list.
Examples of the ARIS modules are the ARIS Suite[282], its components and the extra add-ons being offered for the ARIS tool.

3.8.5 Conclusion

According to GERAM, modules are products that are standard implementations of components, which are likely to be used in the enterprise integration project or by the enterprise itself. Enterprise modules may be configured to form more complex modules, for the use of an individual enterprise. They may be operational processes modules, technology modules or human professions modules, developed either in-house (developed by the architecture frameworks' authors) or by third-party organisations. The enterprise architect / modeller may use trusted modules in order to simplify, speed up and even standardise the modelling task at hand. The architecture frameworks may provide reference models that can be implemented into modules. Alternatively, they may only provide requirements / recommendations for such modules. In-house tools may also be provided as modules - as a whole set, or as individual add-on components[283].

---

[280] such as e.g. Popkin Software's System Architect or Ptech's FrameWork, both described in Section 3.9.2.4.

[281] such as e.g. Popkin Software's Systems Architect C4ISR option, or Ptech FrameWork's 'Military Information Architecture Accelerator', both described in Section 3.9.2.5.

[282] the ARIS proprietary set of modelling tools, described in Section 3.9.2.6 and pictured in Fig. 3.67

[283] the dominant requirement on such tools must be that the set of-, or the add-on components must be *integrated* according to a well-defined meta-model.

3.9 Enterprise Engineering Tools

As previously shown, enterprise modelling frameworks and methodologies employ various approaches (such as using views) in order to limit the size and complexity of the models produced. Commonly however, the resulting models are still too complex, thus requiring computerised enterprise engineering tools.
There are many possible requirements for such tools. They should be able to construct, store, analyse and communicate models. Furthermore, enterprises are moving targets and therefore the tools should be able to (a) promptly build models that reflect the AS-IS state of the modelled entity (before its AS-IS state changes considerably) and (b) update the models as the modelled target evolves. Vernadat (1996) lists additional requirements, such as model archival and management and model execution. As shown in Section 3.7, reference (or partial) models are important as potential resource savers and in preserving the enterprise know-how. Therefore, the modelling tool should support the storage and administration of such models (in the form of templates, libraries, etc). *Executable* models should allow at least (a) model simulation (employ various what-if scenarios towards e.g. optimisation of the business processes) and (b) model-based control of the enterprise (which is a powerful feature, however not always usable for some - e.g. human - aspects of an enterprise).
ISO/TC184/SC5/WG1 (2000a) defines additional requirements for modelling tools, such as:

- the need for a shared design repository, including both formal models and informal descriptions[284];
- ability to operate and manage enterprise operations (e.g. for model-based control)[285];
- ability to connect to the operating environment (where the real business processes reside) in order to update the model;
- modularity and extensibility (including meta-modelling).

Very few existing tools address most of the above requirements[286]. Therefore, the enterprise modeller must make an informed selection of tools, appropriate for the specific modelling task. Reference architectures would usually not

[284] the shared model repository is a current trend of the enterprise engineering / modelling tools. The repository varies in complexity from simple storage structures, to dedicated database management systems and up to 'knowledge' bases that support expert system inference engines. Several such repositories have been briefly presented as part of the reviewed modelling tools in this Section.

[285] this requirement for the modelling tools owes to the possible use of the engineering / modelling tools in both enterprise engineering and management, e.g. as described by CIMOSA.

[286] many of the existing tools are not 'aware' of the model they are creating i.e. they are not based on a metamodel describing the modelling language used (if one is at all available) and hence cannot check the syntax / semantics of the language in question.

prescribe any specific tools - however they often specify requirements and/or recommendations for suitable modelling languages - which in turn are supported by various existing tools. Some modelling tools have been designed to be 'compliant' with a specific architecture - meaning they provide a degree of coverage for the requirements of the particular architecture they comply with. The GERAM metamodel shown in Fig. 3.1 may be used to establish the degree to which a particular tool is compliant with the architecture framework it supports. For example, the tool should support methodologies (i.e. be prepared to manage views and deliverables that the methodology proposes to utilise), implement modelling constructs (language), support generic modelling concepts (i.e. the tool's repository should be based on an integrated metamodel) and be able to store / customise partial models of the supported architecture framework (which may need additional language constructs, as for example discussed in Chapter 17).
This section will only give a basic guidance as to what type of tools may be usable for which architectures. One must remember though that for a typical enterprise modelling task it is almost certain that a combination of languages will be needed. This in turn implies the use of more than just one modelling tool (and the inherent difficulty to manually manage the consistency and cross-relationships between models).

3.9.1 Tools Suitable for Several Architectures

These tools implement one or more of the modelling languages mentioned as usable for various architectures, such as Entity Relationship, the IDEF family of languages, UML, Graphs, etc (refer Section 3.5 for usable modelling languages). Better tools from this category have mechanisms for checking/enforcing the syntax of the language, library maintenance (e.g. for storing partial models), semantics[287] and meta-modelling[288] capabilities. Other such tools are simply specialised graphical editors, providing symbol libraries for the modelling languages they support[289].
The selection of tools suitable for a modelling task essentially depends on the intended life cycle phase coverage of the architectural products. Usually modelling tools feature one or more collections of constructs, which do not necessarily form a proper language. Therefore one should identify the most likely necessary constructs / language and based on those needs then select a preliminary set of tools.

---

[287] i.e. the tool is aware of the *meaning* of the modelling language (if it is based on the ontology of the particular modelling language).

[288] whereby the user may alter the existing metamodel (effectively creating a new modelling languages).

[289] often tools attempting to cover a large range of modelling languages end up being graphical editors with a rich symbol library but no underlying metamodels, i.e. no modelling language *awareness*.

For example, in the Identification phase one may use hand- or computer drawn graphical symbols[290], while in the Implementation phase use of formal languages may be desirable, e.g. if the end products are executable models. Alternatively, use of a word processor, together with a well-defined and consistent glossary and optionally a partial model (e.g. standard / policies manual) may be enough to 'implement' instructions for a human.
In conclusion, a large number of modelling tools support a set of modelling languages in preference to a specific architecture framework. These tools are usable whenever it is possible to use one or more languages they support.
A complete review or list of this type of tools, available or emerging, is beyond the scope of this Chapter. Some such tools are: Knowledge Based Systems' (KBSI) suite of IDEF-based tools (AI0Win, SmartER, ProSim, ProCost, ProABC, etc), Meta Software's Design/IDEF and Design/CPN (for Colored Petri Nets), Computer Associates' ERWin / BPWin tools (for IDEF0/IDEF1x/IDEF3) and Rational Corporation's Rational Rose UML-based modelling tool. Modelling tools supporting particular architecture frameworks are described in Section 3.9.2.

3.9.2 'Compliant' Tools

These tools provide limited support for the methodology and languages employed by the architecture framework they support (i.e. they 'comply with'). Some of the organisations developing the reviewed architecture frameworks have an associated company, which looks after the commercial aspect of the architecture. In such cases, it is this commercial spin-off that develops the architecture framework-compliant enterprise engineering tools (and often also the modelling methodologies)[291]. Other compliant tools have been created by third-party developers (in which case the degree of compliance may be somewhat lower). Finally, several tools have been developed to various stages of completion within research consortia or academia (non-commercial) and are therefore usually openly available.

3.9.2.1 PERA

As previously shown, PERA only provides some recommendations regarding suitable enterprise modelling tools. The PERA guidelines are high-level and therefore they need to be expanded / customised in order to match the modelling task(s) at hand. Once this is done, the users of the PERA framework

[290] for example Rich Pictures. **NB** use of predefined symbols may be good in improving the communication but may also stifle creativity in the early Identification phase. This is particularly true when in the case of graphical editors with predefined and hard to customize graphical libraries, which may distract the user and/or impede lateral thinking.
[291] tools thus developed are called 'in-house tools' in this Chapter.

make their own choices regarding the suitable suite of modelling tools to be employed.

3.9.2.2 CIMOSA

At the time of writing, there were no established in-house tools for CIMOSA. Several third-party CIMOSA-compliant tools do exist however, such as FirstStep, from Interfacing Technologies.

FirstStep is based on three main modules - Designer (model, simulate, analyse business processes), Viewer (distribute, validate, test) and Charter (create process models) (Levi and Klapsis , 1999).

Following an increasingly widespread trend in modelling tools architecture, FirstStep is built on a common repository, which stores all objects used to construct the models. The common repository allows the reuse of the objects while also helping to enforce consistency across the models created in FirstStep.

In addition to the essential modelling and mapping tools, FirstStep also provides a library of partial models specific to various industries, called business templates. These templates may be further specialised and instantiated into the desired model.

CIMOSA aims to produce formal, executable models, which may be used for analysis (simulation) or operation (via model-based control). Taking a similar approach, FirstStep provides a simulation feature, which uses discrete events and probabilistic models of sequential- and concurrent processes.

Again similar to CIMOSA, FirstStep perceives the business process as the core concept in enterprise modelling. Business processes in FirstStep are complex constructs, having essential features such as targets, boundaries, and clients, triggering events, inputs and outputs. The objects used by FirstStep to model business processes reflect the CIMOSA views: Processes, Activities[292] (Functional), Materials / Information (Information), Resources units (Resource) and Organisational units (Organisation).

As previously shown in Section 3.5.3, the CIMOSA languages defined for the views are mutually supporting, e.g. Resources present in a Functional view may have attributes described in the Information view. The same applies for the FirstStep tool, where all objects belong to the shared repository. Thus, an object may appear in several views, each view displaying only the attributes relevant for that view. Processes in FirstStep are linked to resources, departments, business rules, etc creating a holistic representation of the enterprise rather than merely an abstract workflow diagram.

Figure 3.63 shows the relation of the FirstStep to the GERAM (Appendix A of ISO/IS 15704) and ISO/IS 12204, 'Modelling Constructs for Enterprise

[292] in FirstStep, activities are considered to be components of processes. An alternative perception of activities is as a process specialisation, whereby additional attributes or characteristics may be specified (for example temporality).

GERAM	FirstStep	CIMOSA
Function	Process View (static). Process View (dynamic)	Function View (static). Function View (dynamic)
Information	Material view	Information view
Resource	Resource view	Resource view
Organisation	Organisation view	Organisation view
ISO12204	**FirstStep**	**CIMOSA**
General definitions		
Not defined	Modelling View, Template View	Engineering environment, Operation environment
Modelling contructs – Function, view, static		
Enterprise Activity	Scenario Activity Types (6)	Domain Enterprise Activity (28) (Functional Operation)
Modelling contructs – Function, view, dynamic		
Business Process, Event, Seq. Relationships	Process Receive (Process Entries) Event Profile Generator (Resource Links)	Process Event (Behavioural Rules)
Modelling contructs – Organisation view		
Organisational Unit	Organisation view	Organisation cell, Organisation Unit (Organisation Element)
Modelling contructs – Information view		
Enterprise Object, Product, Order, Object View, Relation	Material, Forms	Enterprise Object Object view (Info element) Relation
Modelling contructs – Resource view		
Capability Set, Resource	Active Resource Passive Resource	Capability Set, (Capability) Resource / Func Entity, (Resource Component)

Fig. 3.63. Relation of GERAM and ENV12204 modelling frameworks to FirstStep (based on (Levi and Klapsis , 1999))

Modelling'. FirstStep supports the CIMOSA pre-defined activity types but does not fully model the CIMOSA domain concept. The CIMOSA events are modelled via an activity type (Receive). However, the information modelling capability of FirstStep at the time of writing is still limited[293]. In conclusion, apart from a few differences, FirstStep is a CIMOSA-compliant modelling tool with partial model archival capabilities (as required by GERA).
For more information on the FirstStep modelling tool refer to the Interfacing Technologies web site[294].

3.9.2.3 GRAI

An in-house developed modelling tool for GRAI is IMAGIM from GRAISoft[295]. IMAGIM is a web-based software product built on a modular, client-server architecture. IMAGIM supports enterprise modelling according to the GRAI modelling formalisms and the GRAI methodology. IMAGIM stores the models developed in a GRAI study in a shared repository - a tree-like structure that enforces consistency across models created in IMAGIM and also allows navigation between the models when they are linked (as they share common constructs). The information in the shared database is stored according to a generic GRAI reference model[296].
IMAGIM is composed of a 'kernel' and additional modules. The kernel supports the basic modelling language constructs recommended by GRAI-GIM, as shown in Fig. 3.34 and Fig. 3.36. As such, it provides for modelling of the following systems defined by GRAI:

- the functional and physical systems are modelled using Actigrams (similar to IDEF0 activity diagrams);
- the models of the decisional system are obtained by using the GRAI-Grid (to identify the decision centres, frameworks and flow of information) and the Grai-Nets constructs (to subsequently detail the decision centres);
- the information system is modelled using the Entity Relationship data model.

The IMAGIM kernel allows the creation of the basic GRAI models and also provides for model management (creation / storage / update). Additional modules are available in addition to the IMAGIM kernel. Some modules provide for extended methodology, model and modelling skills management.

---

[293] FirstStep has no ER or equivalent modelling capability. It only allows to name objects (called 'document' in FirstStep) and numbered states of these objects (no attributes, relations or constraints)

[294] at the time of this writing, http://www.interfacing.com.

[295] IMAGIM has been initially developed within the Eureka TIME TOOL project (Doumeingts et al. , 1999).

[296] the generic GRAI reference model "defines the structure and gives an integrated view of the company" (GRAISoft , 2002)

Other modules offer the possibility to enrich the models produced by the kernel or create new models by employing other GRAI methodologies. As several of these modules are still under development (Doumeingts et al. , 1999), only a brief description will be given for each.
The GRAI Study Management module provides for the management of GRAI projects and report generation.
The process module (PROCESSUS) provides several additional features. It allows business process modelling - including temporal aspects (such as sequence links) and automated report generation from the business process models. In addition, the models' performance may be analysed via simulation performed on various criteria (time/cost/quality).
As shown in Section 3.6.2, Performance Indicators Systems may be defined and implemented in GRAI using the ECOGRAI methodology. This methodology is supported by the IMAGIM ECOGRAI module.

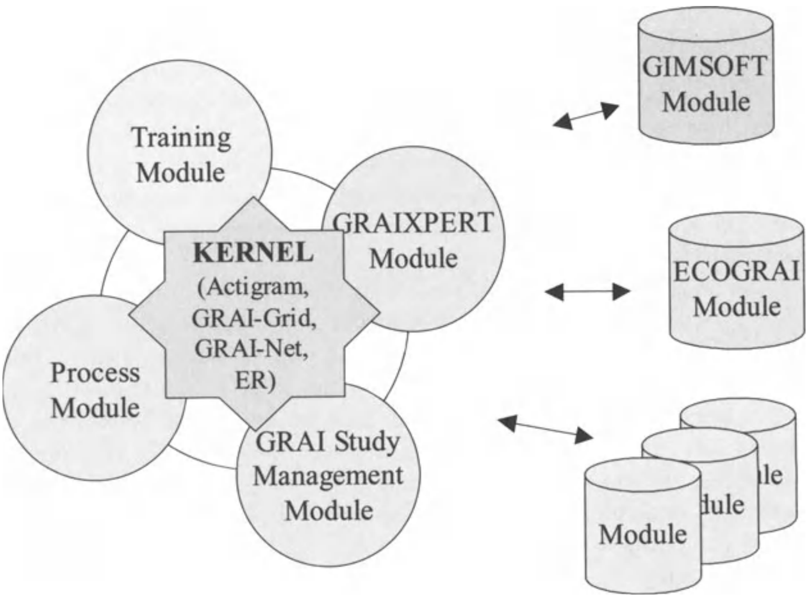


Fig. 3.64. Components of the IMAGIM tool (based on (GRAISoft , 2002)).

Diagnosis of the modelling work and model coherence checking services are offered by a rule-based[297] expert system module called GRAIXPERT.
The GIMSOFT IMAGIM module supports the GIMSOFT methodology, which covers specifications for choosing / implementing ERP / Production Management software packages.

[297] based on ILOG's http://www.ilog.com/ business rule creation software (Doumeingts et al. , 1999).

Finally, GRAI Education provides for training in the GRAI methodologies. The components of the IMAGIM tool are shown in Fig. 3.64. Other methodological modules (e.g. for auditing, business planning, quality assurance, knowledge management, etc) are described on the GRAISoft web site[298].

3.9.2.4 Zachman

At the time of this writing, there are no publicly established in-house Zachman-compliant modelling tools.
Third-party developers have either built their Zachman-compliant tools around the Zachman framework or offer plug-ins for it. Two such developers have been reviewed: Popkin Software and Ptech, Inc.

3.9.2.4.1 Popkin Software's System Architect

Popkin software's System Architect is an 'all-rounder' tool in the sense that it supports a large number of modelling methods. This approach has its advantages (e.g. allowing its use across businesses by personnel trained in various modelling methodologies / languages) and disadvantages (such as not being the 'best' product for any given methodology). System Architect is developed around a shared 'corporate' repository, which supports model-based and project-based approaches. Various modelling techniques may be used with the same repository, enabling model sharing across projects (which may use different methodologies).
As can be seen from Fig. 3.65, System Architect's repository is customisable, i.e. the user may alter the predefined meta-data, including integrity constraints[299]. As previously shown in this Chapter, this is a powerful but also somewhat hazardous feature (tampering with the metamodel essentially changes the meaning of the modelling language, with expressiveness and complexity consequences). System Architect supports most mainstream structured- and object-oriented modelling methods (refer Fig. 3.65), by using an Enterprise Architecture Framework (modelled after the Zachman framework) and providing a modelling methodology of its own (Popkin Software , 2001) partly guided by the Catalyst[300] approach.
System Architect provides support for almost all cells of the Zachman framework, plus the capability to view relationships between several cell contents. A matrix editor allows cross-referencing and correlating the design artefacts across several models. Add-ons (modules) are provided for business analysis, such as Activity-Based Costing (ABC) or model simulation (based on process charts and IDEF3).

[298] at the time of writing, http://www.graisoft.com.
[299] e.g., the user may change the definition of what is a 'legal' model.
[300] an organisation modelling methodology developed by the Computer Sciences Corporation (CSC), http://www.csc.com.

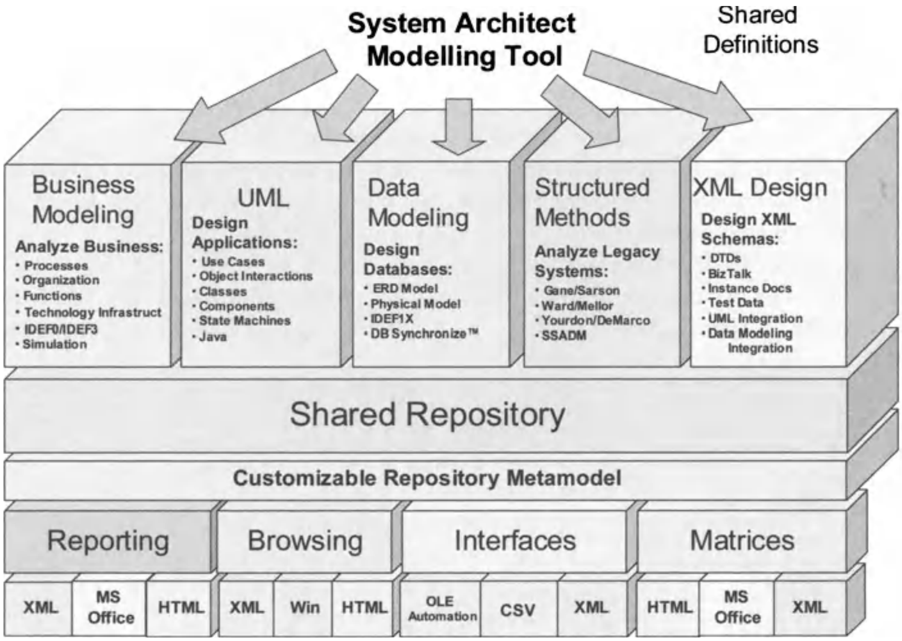


Fig. 3.65. System Architect's Structure (based on (Popkin Software , 2002))

System Architect covers the Zachman framework mainly from a software development point of view. The languages employed in modelling the Zachman framework cells views reflect this orientation (refer Fig. 3.39). Therefore, while in essence enterprise architects (and not just software engineers) may model a business using the System Architect tool within the specifications of the Zachman framework, attention must be paid to the proper modelling of the decisional, human and social aspects of the enterprise. System Architect remains a good cross-modelling methodologies tool, using the Zachman framework as its architecture framework.

3.9.2.4.2 Ptech's FrameWork

Ptech, Inc has developed a methodology-independent integrated modelling environment called FrameWork. It is based on an object-oriented data structure which has its semantics fully defined via metamodels. Being methodology-neutral, FrameWork is claimed to have wide applicability (e.g. be able to capture, analyse and design data and link it to organisations, activities, locations, etc.). The downside to its genericity is that it needs to be specialised for particular modelling tasks via specific extensions called 'accelerators' (John , 2001). This is resolved by use of plug-in modules for specific architectures, which provides a consistent extensibility mechanism for this modelling tool.

FrameWork is built on the Object-Oriented paradigm without necessarily using an Object-oriented notation[301]. Since all data is stored in a common database, all definitions / attributes and models are consistent across various representations. FrameWork supports user-defined queries and outputs, as well as interfaces to other products via standard data formats (e.g. (e.g. Comma Separated Value (CSV)). It also offers a 'customisation kit', which allows the user to define custom model types (i.e. modify existing metamodels) and modelling tool behaviour.
FrameWork offers full support for the Zachman framework[302], which is used in preference to organize enterprise architecture information. Ptech also offers a modelling methodology associated with their tool, called Causal Architecture (briefly described in Section 3.6.4).

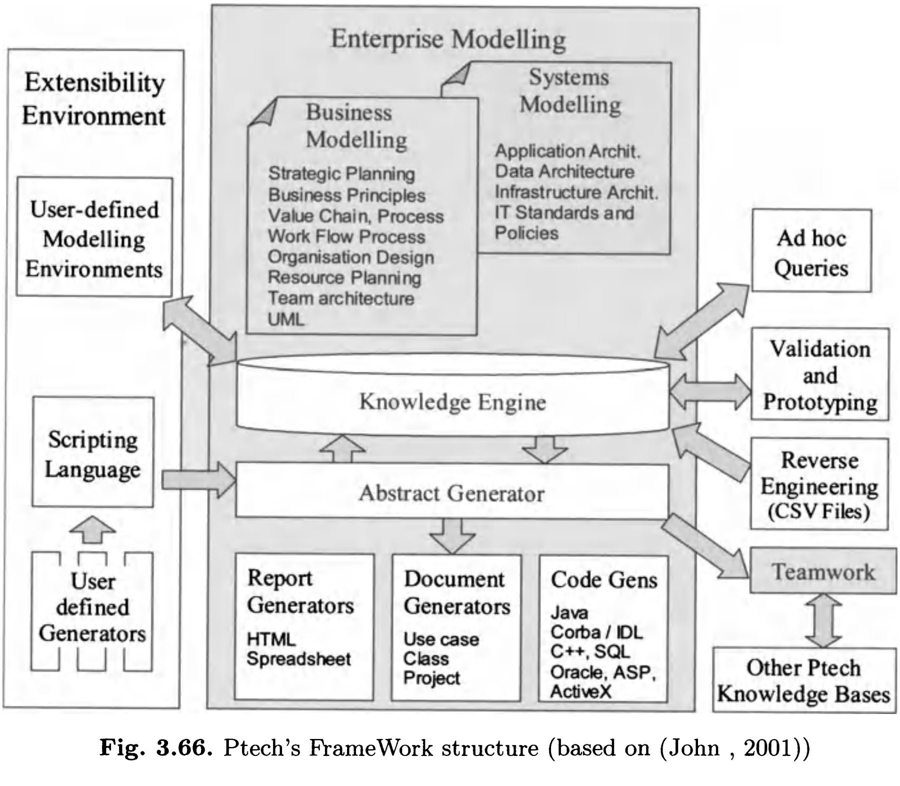


Fig. 3.66. Ptech's FrameWork structure (based on (John , 2001))

FrameWork is based on a shared knowledge base designed as a semantic network, which is also a rule-based inference tool. This design enforces consistency

[301] e.g., the user may choose SADT notations in preference to UML.
[302] via the Zachman framework Control Panel, containing a template and most of the model types required.

of existing and newly added objects. Owing to this structure (refer Fig. 3.66), FrameWork may represent the same data in a Zachman framework-based view (based on Zachman perspectives and abstractions) or from a C4ISR viewpoint[303].

3.9.2.5 C4ISR

At the time of this writing, there are no established in-house tools available for C4ISR. It must be noted though that a Joint C4ISR Architecture Planning / Analysis System (JCAPS II) is being developed, comprising a shared repository of architectural products and associated web-based services to create / modify / view / publish products and data. JCAPS II will be built on top of a CADM- compliant database[304] and it is intended to be tool-independent[305]. The nature and extent of the use of C4ISR framework[306] have prompted third-party software developers to start offering add-on modules supporting C4ISR for their existing tools. At the time of this writing, such developers are Popkin Software and Ptech, Inc.

3.9.2.5.1 Popkin Software System Architect's C4ISR Option

System Architect has previously been generally described in Section 3.9.2.4.1. Popkin Software has developed a C4ISR add-on module for their modelling tool. The module is based on a repository described in the AV-2 (Integrated Dictionary). The C4ISR add-on supports all the C4ISR architectural views and provides automated report generation for the C4ISR applicable deliverables[307]. It also appears to offer limited support for the C4ISR partial models, such as the System Evolution (SV-8, derived from the repository), System Forecast and Technology Forecast (SV-9 and TV-2). There is however no explicit support for the projected JCAPS II or for the C4ISR data model[308], although the transfer of data between systems is potentially possible using System Architect's export facility via standard formats. The main strength here is the common underlying shared repository ensuring the consistency of C4ISR deliverables.

[303] the validity of the mapping thus obtained depends essentially on the underlying metamodels. Therefore, the user is advised to test such mappings prior to employing this tool for e.g. conversion of models.

[304] the original JCAPS was not CADM compliant and did not support the Global Information Grid (GIG) or the C4ISR Joint Operational Architecture (JOA) efforts (Levis , 2001a).

[305] notwithstanding this, the JCAPS II initiative of the US Department of Defence (DoD) is already influencing the development of C4ISR-compliant tools

[306] The reader is reminded that C4ISR is the architecture framework adopted by the US Department of Defence.

[307] such as the Technical Architecture Profile, TV-1.

[308] although this may be made available in the future via a mechanism similar e.g. to the meta-data customisation facility.

3.9.2.5.2 Ptech FrameWork's Military Information Architecture Accelerator

For a general description of Ptech's modelling tool refer Section 3.9.2.4.2. Ptech, Inc. has developed a FrameWork Military Information Architecture (MIA) plug-in[309], for the C4ISR architecture framework. The plug-in supports all C4ISR's architectural products and a few additional services, such as architecture management (e.g. versioning), interfacing to schema-driven tools, scenario analysis, etc.
Importantly, the MIA plug-in allows all architectural products to be linked to a unique underlying database, thus ensuring the consistency of the models produced (a much needed feature in C4ISR, due to the lack of an established modelling methodology, as earlier shown in Section 3.6.5). Reports for architectural products such as the Architecture Summary and Information (AV-1), Technical Architecture Profile (TV-1) and Standards Technology Forecasts (TV-2) may be automatically generated from the underlying data (where applicable). An important feature is the automated generation of the C4ISR Integrated Dictionary (AV-2) in a web-enabled and illustrated form, based on the common architecture data.
The MIA plug-in provides for scenario analysis and validation, but not for e.g. the generation of executable models for performance evaluation. Interfaces are also available for potential architecture planning and analysis systems (such as JCAPS) in a basic format (using CSV).
The C4ISR plug-in module also has limited support for the C4ISR partial models. For example, it supports system evolution (SV-8), system- and technology forecasts (SV9 and TV-2) and partially the Technical Reference Manual (TRM), but offers no explicit support for e.g. interoperability levels (LISI), joint task lists (UJTL) or shared data environments (COE, SHADE). It is also not explicitly stated whether the C4ISR data model (CADM) is represented, possibly as a specialised repository schema[310].

3.9.2.6 ARIS

ARIS emphasizes the role of adequate enterprise engineering tools in the success of an architecture framework. Thus the in-house ARIS Collaborative Suite has been developed in order to provide modelling, analysis and review of the business processes. It supports the ARIS methodology (Scheer , 1992) and as of version 6 consists of the Web Designer, Toolset and Easy Design. The three main components share core components and capabilities such as the Explorer (model navigation), the Designer (modelling), editing of database tables (direct / bulk data input / change) and multilingual interfaces. As can

[309] Ptech labels the commercial plug-ins for their FrameWork "accelerators' (refer Section 3.9.2.4.2.

[310] in the author's opinion, the main knowledge base is method independent and thus more general than the C4ISR CADM. However, once the knowledge base is specialised for a C4ISR architecture it should be represented via the CADM.

be seen from Fig. 3.67, the Web Designer is Java-based and therefore intended to be platform-independent (usually at the cost of processing speed). The rest of the ARIS suite components are platform-specific.

The ARIS tool is built on top of a shared relational database, with all information being stored as user-editable tables. As expected, this architecture enforces consistency, enables reuse of objects across models while also enabling model linking via shared objects. The database foundation also allows formulation of queries, generation of reports and tuple extension - i.e. further properties (fields) may be added to an object stored as a tuple.

The ARIS models are created at the type level but stored and administered at meta-meta level ((Scheer , 1999); refer Fig. 3.30). This approach aims to ensure that the objects stored are independent of the modelling methods. Therefore the user may change the metamodel by altering-, removing or adding to the instances (records) in the meta-metamodel table[311].

The three main components of ARIS are aimed at different groups of users within the enterprise, but in their common features they all aim at creating a set of models reflecting ARIS concepts (data, function, process, organisation, output).

ARIS Toolset is (traditionally) the main component of the suite, encompassing the full set of design features. It is aimed at process managers / owners and business analysts. The Toolset component may be also used for server- and business cases management. It is more customisable than the other modules and it also allows to create model variants, which may remain linked (or traceable) to their original model.

Easy Design is aimed towards people in the enterprise with little training in business process modelling, but who own the *knowledge* that needs to be captured in the model(s). Therefore Easy Design only includes the methodical scope for common use cases. Easy Design may also be used to implement or convert legacy databases, including semantic checks to check models consistency.

ARIS Web Designer features a browser-enabled front-end, therefore being independent of platform or geographic location. It targets both those new to business process design and process managers, which may need global access to the information.

ARIS provides add-on modules that may be called upon by the three main ARIS components[312]. These modules provide additional functionality such as activity-based costing (ARIS ABC), model simulation (ARIS Simulation), interfaces to other packages (e.g. SAP, Lotus Notes), web connectivity, etc. These add-on modules are Enterprise Modules in the GERAM sense since

[311] for example, if a new modelling method requires a new type of construct (such as enterprise entity type, as shown in Section 3.8.1.6), the object in question will simply be added to the meta-metamodel table (which stores object types).

[312] ARIS Toolset, as the most comprehensive ARIS Suite component, can access the full set of add-ons. The other main ARIS components may only call selected add-ons (relevant to their restricted scope).

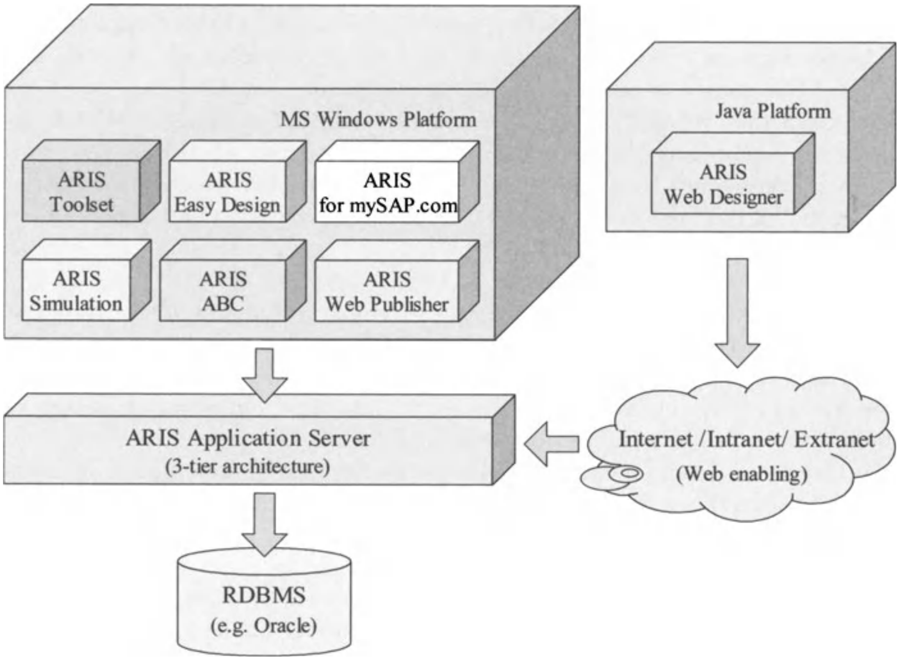


Fig. 3.67. The ARIS Collaborative Suite Components (based on the structure shown in (IDS Scheer , 2001)).

they provide out-of-the-box functionality, which may be employed to construct Enterprise Operational Systems (EOS). In a broader context, the ARIS Collaborative Suite itself may be considered a kind of Enterprise Module in the GERAM sense (refer Fig. 3.1)[313].

3.9.2.7 Conclusion About Enterprise Engineering Tools

Enterprise modelling tools provide capabilities that can be employed to create enterprise models, which in turn may be used in order to ultimately implement Enterprise Operational Systems. Thus, modelling tools (as a whole, or separate plug-and-play modules for them) are in fact a special type of enterprise module, as defined in GERAM. Similar to the modelling languages, there is no 'complete' enterprise modelling tool suitable for modelling all aspects of an enterprise. As shown in Section 3.4.7, some of the reference architectures reviewed do cover the complete GERA scope, but each goes into various degrees of detail, from different aspects. Therefore, a tool claiming 'compliance' with a particular architecture framework would still potentially cover only

[313] this applies to all modelling tools that meet the GERAM requirements for an Enterprise Modelling Tool.

some of the modelling aspects needed for a particular (or type of) enterprise integration task.

At the time of this writing, some common trends in modelling tool development are web-enabling, the use of a central repository and modularity (plug-in modules). The approach to the repository varies from a plain (mostly provided by a third party, such as e.g. Oracle) relational database, to knowledge bases featuring inference engines. Most of the tools strive to provide *meta*modelling - an advanced feature, which however may open a floodgate to uncontrolled modelling language proliferation. The plug-in modules are provided for further processing (e.g. simulation) of the models produced, for modelling additional aspects of the enterprise, or for interfacing with other established (e.g. Resource Planning, etc) tools. The term 'Web-enabled' used by the tool developers usually covers front-ends, on-line collaboration, publishing, etc.

In conclusion, the answer to an 'all rounder' modelling tool may well be a *meta*-tool, describing the rules of how to combine the available modelling tools for a type of modelling task[314]. Such a tool would sit on top of existing modelling tools (rather than coercing these tools to comply with it) and could possibly implement constructs contained in a meta-language[315].

A comprehensive review of commercial third party and proprietary modelling tools is beyond the scope of this Chapter. Enterprise modelling tools are perhaps the most dynamic (and therefore least predictable) component of the GERAM. They are continuously affected by the rapid evolution of software and hardware platforms and the ever shorter software / hardware product update cycles. Therefore a complete and up-to-date review would be more suitably presented in the form of (regularly updated) website content, rather than in printed form[316].

3.10 The Big Picture and Conclusions

3.10.1 The Big Picture

The effort of mapping the major architecture frameworks against a fixed reference (GERAM) has validated most of the results of previous work but has also provided new insights into the frameworks in question and into architecture frameworks in general. A synthesis of the most important findings is

[314] such a meta-tool may e.g. be an expert system that designs a modelling tool / language selection solution on the fly for a particular modelling task. The design principles of such a tool are being researched at the time of writing in a PhD program within Griffith University.

[315] although not explicitly defined as a meta-language as such at the time of this writing, this may possibly be the UEML. UEML attempts to provide an unifying umbrella (based on clear ontologies and metamodel) over existing languages rather than enforcing yet another modelling language (Vernadat , 2001).

[316] the printed form may be more stable but only offers a state-of-the-art at a given point in time, potentially obsolete by the time the print is made available.

presented in Section 3.10.2. The author has attempted to keep an open mind and identify the contribution made by each of the reviewed frameworks to the field of enterprise integration, rather than trying to identify a single 'best' architecture framework. A 'best' architecture framework for all modelling purposes does not and may never exist, since the human mind will always find new aspects to investigate and create models from new perspectives.
For this reason, the enterprise architect will have to select a set of existing architecture frameworks in order to cover all necessary aspects of the enterprise. The selection should be the result of matching the particular enterprise modelling requirements against the currently available modelling artefacts. The client (such as the management of the enterprise in question) must also be aware of the *actual* modelling needs of its enterprise, enabling proper negotiation with the (in-house or outsourced) providers of engineering / modelling services. This book (and implicitly this Chapter) aims both to provide this much needed knowledge to the client and at the same time to be a guide for enterprise architect(s). Management may thus take initial, preliminary decisions 'in the know' regarding the necessity, extent, scope, timing, etc of the modelling effort without necessarily involving specialised (and potentially biased) third-party services. Such services may subsequently become necessary once the GERA Identification and partly Concept phases are accomplished.
Architecture frameworks will evolve and new frameworks will certainly emerge (or existing ones will gain more recognition). Therefore, it is important that the common reference used in the mapping endeavour reflects the state-of-the-art in the domain and proactively provides for any identified future trends. GERAM is no exception - hence Chapter 2 of this book.

3.10.2 Final Conclusions

GERAM provides a generalised framework against which the contents of the various modelling frameworks (reference architectures, modelling methodologies, languages, etc) may be mapped. This framework is void of content and as such *not* a competitor of any of the reviewed and/or mapped architectures. It is however very useful as a common baseline[317] and to-do list for a type of- or specific enterprise integration and modelling task.
The same artefact may play different roles within an architecture framework. For example, the PERA diagram (a model of the modelling methodology described in the PERA Handbook (Williams et al. , 2001) is describing the necessary steps towards enterprise engineering and the deliverables involved at each stage in detail. Therefore, it is both a life-cycle architecture (describing the integration process) and a modelling framework[318].

---

317 and as such promote consensus among the various enterprise integration research directions (Bernus and Nemes , 1997).

318 A *generalised* modelling framework, such as the one contained in the GERA reference architecture, holds *requirements* for models and also serve as a checklist

A corollary of the above is the fact that artefacts relevant to the proposed mappings to GERAM (such as life cycle phases, life history, views etc) may reside in *any* component of the architecture framework in question. It has been thus possible to map life cycle phases present e.g. within an architecture framework's *methodology* (rather than in its modelling framework) to the GERA life cycle phases[319].
In this context, the difference between the architecture framework's components is often blurred. Just consider the fact that an architecture may also be considered a reference model or that a modelling framework populated with content (models) is potentially a language in itself.
The modelling frameworks of some life cycle architectures allow placeholders for more than just model constructs. Prime examples are CIMOSA and GERA with their explicit modelling framework genericity dimensions, which provide for partial models and metamodels, glossaries, ontologies, etc.
The decisional aspect of enterprise modelling is perceived in different ways by the modelling frameworks of the reviewed architectures. While some cover the decisional aspect separately and in great detail (e.g. GRAI), others consider the decisional aspect as part of the organisational- (e.g. partly CIMOSA) or functional (e.g. ARIS) views. The GERA approach is neutral, in the sense that it allows the decisional view to be represented in various views as long as the necessary degree of detail is covered. In the opinion of the author, the views contained in GERA (such as Functional, Information, etc) display various degrees of independence from one another and may (implicitly or explicitly) contain other aspects. For example, the Organisation view in GERA is understood to represent 'who does what', i.e. the mapping of the Resources to the Functions[320]. Moreover, in the opinion of the author the Decisional aspect may be considered a specialisation of the Functional view - therefore for example a Decisional Centre (DC) in the GRAI sense may well be e.g. a Functional Entity in the CIMOSA sense.
Reference models may be obtained using a top-down approach, i.e. derived from ontologies and metamodels and specialised to the desired level. However, a bottom-up approach may also be employed, whereby particular case studies may be grouped together (e.g. in classes) by various criteria and a partial model (class model, in object-oriented speak) may be inferred from them. The partial model obtained through the top-down approach must be validated by several particular models (instantiations of the partial model). The bottom-up approach has the advantage of already being validated by the particular models used in the inference process.

---

as to what models are needed for an unambiguous representation of the enterprise entity.

319 refer e.g. the GRAI-GIM mapping shown in Fig. 3.3 or the C4ISR mapping shown in Fig. 3.8

320 this perception of the Organisation view holds true for both human and machine Resources - therefore implying that Organisation may refer to both human and non-human (e.g. software agent) aspects.

A common occurrence among the reviewed architectures is the absence of an explicit concept for a corresponding GERA construct. However, most of these architectures do provide constructs that cover at least part of the missing concept. Examples of such constructs:

- the concept of life cycle is not explicitly stated, but several life cycle phases or equivalent are provided (e.g. the Zachman or C4ISR views);
- the genericity dimension is supported indirectly (e.g. by PERA, refer Fig. 3.18);
- the life history concept is indirectly supported (e.g. in CIMOSA via the System Change and Maintenance phase and in C4ISR via transition views).

An important life cycle phase, often overlooked (or at least not fully covered) by the reviewed architectures is *decommissioning*. This is a crucial phase in view of knowledge- and other resources' preservation and reuse. PERA for example provides good coverage of the decommissioning (called Obsolence), followed either by Revamping (involving reiteration of one or more life cycle phases) or by Dissolution.
Mapping the reviewed architectures against a common *reference* architecture (such as GERA) has the benefit of providing some clarification in relation to the various meanings attached to terms commonly used, or implied in these architectures. Examples of such terms are 'view'[321], 'life history' and 'horizon'[322];
Some general concepts are present in every architecture framework - whether explicitly covered or not -, such as versioning (for example, explicitly covered in Zachman as a form of (temporal) recursion).
Matrix, three- or multidimensional[323] frameworks try to maximise their expressive power by offering all possible combinations of views. This is feasible only in the context of filtering out the view combinations that prove to have irrelevant content for the type of modelling tasks that the given framework endeavours to support.
Proprietary methodologies may provide commercial advantages, but due to their closed nature they rarely advance the cause of enterprise modelling as such and do little to provide public interest for the reference architecture or architectural deliverables they support. A possible solution may be a mixture of publicly available white papers (laying the foundations (e.g. metamodels, ontologies) for the methodologies and providing the basis for public debate /

[321] used as such (view) in GERA, CIMOSA, IMPACS / GRAI-GIM, possibly meaning 'life cycle phase' in C4SIR, or called 'perspective' in Zachman.
[322] used with different meanings in e.g. GRAI Grid vs. C4ISR (Strategical, Tactical, Operational, etc)
[323] many frameworks go beyond the geometrical three-dimensional limit and define additional implicit dimensions. For example., GERA's life history concept relies upon a *time* dimension, which is however best represented separately from the GERA 'cube'.

acceptance / validation) and proprietary detailed methodologies for commercial use.
Historically, most architecture frameworks came into existence in an initial incomplete form[324] and have subsequently (subject to their envisaged purpose) evolved into a more complete framework. This evolution process is continuous but not always visible, also due to proprietary and commercial issues. In any case, GERAM may also assist the evolution of such life cycle architectures by identifying gaps in their frameworks and helping define their desired future development areas.
The architecture frameworks involved in the development of GERAM have already benefited from their participation by achieving a better understanding of their own structure and their contribution to the overall enterprise integration endeavour. The dialog and exchange of ideas within the IFIP-IFAC Task Force[325] on Architectures has not only resulted in a commonly accepted set of requirements for architecture frameworks ISO/TC184/SC5/WG1 (2000b) but has also promoted a synergy towards advancing the cause of enterprise architectures. A continuation of this effort (involving all major architecture frameworks) is necessary in order to ensure that GERAM stays up-to-date as the enterprise integration domain matures.

3.11 Glossary of Terms Used in this Chapter

This section lists abbreviations and special meanings attached to terms, which are valid only within this Chapter (all abbreviations have been also fully stated at their first occurrence in text).
Abbreviations:

- ADF: Australian Defence Force
- AF: Air Force
- ARIS: Architektur für Informations Systeme (Architecture for Information Systems)
- C4ISR: Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance
- CIMOSA: Computer Integrated Manufacturing - Open System Architecture.
- CSV: Comma Separated Value
- DoD: (United States) Department of Defence
- EBE: Enterprise Business Entity
- GRAI/GIM: Graphes avec Résultats et Activitées Interreliées (Graphs with Results and Activities Interrelated)/ GRAI Integrated Methodology

[324] e.g. GRAI as a collection of methodologies, Zachman - as a partial modelling framework, CIMOSA - focused on the functional / behavioural and formal / executable aspects, etc.

[325] (IFIP-IFAC Task Force , 1993)

- HQ: Headquarters
- IEM: Integrated Enterprise Modelling
- OSTN: (IDEF3) Object State Transition Networks
- PERA: The Purdue Enterprise Reference Architecture
- STD: State Diagram
- UML Unified Modelling Language
- UOB: (IDEF3) Unit Of Behaviour

Terms:

- Architecture Framework: Life cycle architecture as a whole, comprising at least a modelling framework and one or more of the components shown in the GERAM (Fig. 3.1)
- Artefact generalisation of the enterprise concept - in an attempt to extend application beyond enterprise modelling
- Cube (three-dimensional) modelling framework of a life cycle architecture
- In House internal to the organisation (the 'house')
- Instantiation the process of obtaining a particular model out of a partial model by allocating values (or known domains) to all the partial model variables. Also used to obtain a particular instance of a particular model.
- Life Cycle a set of possible phases that an entity *may* go thorugh in its life
- Life History the irreversible succession of life cycle phases that has happened (or is reasonably expected to happen) in the life of an enterprise
- Modelling Framework part of the Reference Architecture, , a systematic presentation of models and their scope relevant throughout the life cycle of enterprise entities
- Reference Architecture: a framework containing a Modelling Framework and other relevant concepts (such as life-cycle, life history and enterprise entity type)
- Specialisation: the process of obtaining a partial model from another partial model by allocating values (or known domains) to some of the partial model variables (and possibly adding new variables)
- Windchime: the PERA diagram

References

Berio, G. and Vernadat, F.B. (1999). Enterprise Modelling with CIMOSA: Models and Methods. Paper presented at the IEMC99 - IFIP International Enterprise Modelling Conference, Verdal, Norway.

Bernus, P. (2000). Review of the State of the Defence Information Environment Architecture, [Report to the Australian Defence Force Headquarters]. School of CIT, Griffith University.

Bernus, P. (1999). What, Why and How to Model In The Enterprise ? Paper presented at the IFIP International Enterprise Modelling Conference (IEMC '99), Verdal, Norway.

Bernus, P. and Nemes, L. (1994). A Framework to Define a Generic Enterprise Reference Architecture and Methodology (MTM366). Melbourne: CSIRO Division of Manufacturing Technology.

Bernus, P., and Nemes, L. (1997). The Contribution of the Generalised Enterprise Reference Architecture to Consensus in the Area of Enterprise Integration. Paper presented at the ICEIMT '97.

Bernus, P., Nemes, L., and Morris, B. (1996a). The Meaning of an Enterprise Model. In Bernus, P. and Nemes, L. (Eds.), Modelling and Methodologies for Enterprise Integration (pp. 183-200). London: Chapman & Hall.

Bernus, P., Nemes, L., and Williams, T. J. (1996b). Architectures for Enterprise Integration. London: Chapman & Hall.

Bernus, P., Nemes, L., and Williams, T. J. (1996c). Mapping against a matrix. In Bernus, P., Nemes, L. and Williams, T.J. (Eds.), Architectures for Enterprise Integration (pp. 281-309). London: Chapman & Hall.

Bienvenu, P., Kim, D., & Levis, H. (2000). C4ISR Architectures III: An Object Oriented Approach to Architecture Design. Journal of INCOSE, 3(4).

Bravocco, R., and Yadav, S. (1985). Requirements definition architecture - an overview. Computers in Industry, 6, 237-251.

C4ISR Architectures Working Group. (1997). C4ISR (v. 2.0). US Department of Defense (DoD). Available: http://www.cisa.osd.mil [1997, 2002].

CEN/TC 310/WG1. (2002). EN ISO/CD 19440 (Revision of ENV12204 - Constructs for Enterprise Modelling) (Unpublished draft version).

Chen, D., and Doumeingts, G. (1996) .The GRAI-GIM Reference Model, Architecture and Methodology. In Bernus,P., Nemes, L., and Williams, T. J. (Eds.), Architectures for Enterprise Integration. London: Chapman & Hall.

Chen, D., Vallespir, B., and Doumeingts, G. (1997). GRAI Integrated Methodology and its mapping onto generic enterprise reference architecture and methodology. Computers in Industry, 33(3), 387-394.

Chen, D., Vallespir, B., and Doumeingts, G. (2002). Designing Manufacturing Systems: Contribution to the Development of An Enterprise Engineering Methodology (EEM) within the frame of GERAM. Paper presented at the IFAC World Congress, Barcelona.

Chen, D., Vallespir, B., Doumeingts, G., and Zanettin, M. (1990). The IMPACS Conceptual Reference Architecture. Bordeaux: Report to ESPRIT, GRAI Laboratory, University of Bordeaux.

CIMOSA Association. (1996). CIMOSA - Open System Architecture for CIM,. Technical Baseline, ver 3.2. Private Publication.

DeMarco, T. (1979). Structured Analysis and systems specification. Englewood Cliffs, NJ: Prentice Hall.

Doumeingts, G. (1984). La Methode GRAI [PhD Thesis]. Bordeaux, France: University of Bordeaux I.

Doumeingts, G. (1998a). GIM, Grai Integrated Methodology. In Molina, A., Kusiak, A. and Sanchez, J. (Eds.), Handbook of Life Cycle Engineering - Concepts, Models and Methodologies (pp. 227-288). Dordrecht: Kluwer Academic Publishers.

Doumeingts, G., Ducq, Y., and Kromm, H. (1999). Enterprise Modelling Techniques to improve efficiency of enterprises. Paper presented at the IEMC '99 - IFIP International Enterprise Modelling Conference, Verdal, Norway.

Doumeingts, G., Vallespir, B., and Chen, D. (1998b). GRAI Grid Decisional Modelling. In Bernus, P.,Mertins, K. and Schmidt, G. (Eds.), Handbook on Architectures of Information Systems (pp. 313-339). Heidelberg: Springer Verlag Berlin.

Doumeingts, G., Vallespir, B., Zanettin, M., and Chen, D. (1992). GIM-GRAI Integrated Methodology - A Methodology for Designing CIM Systems. Bordeaux: Ver. 1.0, Unnumbered Report, LAP/GRAI, University Bordeaux I.

ESPRIT Consortium AMICE. (1993). CIMOSA - Open System Architecture for CIM (2^{nd} revised & extended ed.). Berlin: Springer-Verlag.

Fox, M. S. and Gruninger, M. (1998). Enterprise Modelling. AI Magazine, 19(3), pp.109-121.

Goranson, H. T. (1998). Agile Manufacturing. In Molina, A., Kusiak, A. and Sanchez, J. (Eds.), Handbook of Life Cycle Engineering - Concepts, Models and Methodologies. Dordrecht: Kluwer Academic Publishers.

GRAISoft. (2002). IMAGIM software product, [PDF document]. GRAISoft. Available: http://www.graisoft.com [2002, Aug 2002].

Hay, D. C. (2000). A Different Kind of Life Cycle: The Zachman Framework, [White Paper]. Essential Strategies, Inc. Available: http://www.essentialstrategies.com [2002, April 2002].

ICAM. (1981). Integrated Computer Aided Manufacturing (ICAM) Architecture Part II, Vol IV - Functional Modeling Manual (IDEF0) (AFWAL-TR-81-4023). Wright-Patterson Air Force Base, Ohio: Air Force Materials Laboratory.

IDS Scheer. (2001). ARIS 6 - Collaborative Suite, [White Paper]. IDS Prof. Scheer GmbH. Available: http://www.ids-scheer.de [2002, Aug 2002].

IFIP-IFAC Task Force. (1993). IFIP-IFAC Task Force on Architectures for Integrating Manufacturing Activities and Enterprises. IFIP Newsletter / IFAC Newsletter.

Industry-University Consortium. (1992). An Implementation Procedures Manual for Developing Master Plans for Computer Integrated Manufacturing. Purdue University, West Lafayette, Indiana.

Inmon, W. H., Zachman, J. A., and Geiger, G. J. (1997). Data Stores, Data Warehousing and the Zachman Framework. New York, NY 10011: McGraw-Hill.

Instrument Society of America (ISA). (2000). ISA–95.00.01–2000 Enterprise-Control System Integration Part 1: Models and Terminology. North Carolina 27709.

ISO/TC184/SC5/WG1. (2000a). Annex A: GERAM. In ISO/TC184/SC5/WG1 (Ed.), ISO/IS 15704: Industrial automation systems - Requirements for enterprise-reference architectures and methodologies.

ISO/TC184/SC5/WG1. (2000b). ISO/IS 15704: Industrial automation systems - Requirements for enterprise-reference architectures and methodologies.

Jensen, K. (1992). Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use (Vol. 1). Berlin: Springer-Verlag.

Jick, T. D. (1979). Mixing Qualitative and Quantitative Methods: Triangulation in Action. Administrative Science Quarterly, 24(4).

John, L. (2001). PTech's FrameWork^{TM} Military Information Architecture Accelerator, [Lecture Slides]. PTech, Inc. [2002, Aug. 2002].

Kosanke, K., and Vernadat, F. (1996). Contributions of CIMOSA to GERAM. Paper presented at the IFAC 13th Triennial World Congress, San Francisco, USA.

Kosanke, K., and Vernadat, F. (1998). CIMOSA-Life Cycle Based Enterprise. In Molina, A., Kusiak, A. and Sanchez, J. (Eds.), Handbook of Life Cycle Engineering - Concepts, Models and Methodologies (pp. 181-225). Dordrecht: Kluwer Academic Publishers.

Levi, M. H., and Klapsis, M. P. (1999). FirstSTEP process modeler — a CIMOSA-compliant modeling tool. Computers in Industry, 40, 267-277.

Levis, A. H. (2001a). C4ISR Architecture Framework Implementation, [Lecture Slides]. System Architectures Laboratory, George Mason University, C3I Centre, Fairfax VA [2002, July 2002].

Levis, A. H., and Wagenhals, L. W. (2000). C4ISR Architectures I: Developing a Process for C4ISR Architecture Design. Journal of INCOSE, 3(4).

Levis, A. H., and Wagenhals, L. W. (2001b). Advances in C4ISR Architectures, [Lecture Slides]. System Architectures Laboratory, George Mason University, C3I Centre, Fairfax VA [2002, May 2002].

Li, H., and Williams, T. J. (1994). A formalization and extension of the Purdue Enterprise Reference Architecture and the Purdue Methodology (Report 158). West Lafayette, Indiana: Purdue Laboratory for Applied Industrial Control.

Martin, R., and Robertson, E. (2000). Enterprise Architecture Frameworks. Computer Science Department, Indiana University. Available: http://www.cs.indiana.edu/database/Publications [2001, Nov 2001]

META Group Inc. (2002). Enterprise Architecture Desk Reference, [Executive Summary]. META Group. Available: http://www.metagroup.com [2002, Aug 2002].

Noran, O. (2000a). Mapping of ISO15288 and ISO12207 onto ISO15704 [Master Thesis]. School of CIT, Griffith University. Available: http://www.cit.gu.edu.au/~noran; http://www.gu.edu.au/ins/lils/GriffLink/ [2002, Jan 2003].

Noran, O. (2000b). Business Modelling: UML vs. IDEF, [Report / Slides]. School of CIT, Griffith University. Available: http://www.cit.gu.edu.au/~noran [2003, Jan 2003].

Popkin Software. (2001). Building an Enterprise Architecture: The Popkin Process (v1.0), [White Paper]. Popkin Software. Available: http://www.popkin.com [2002, 2002].

Popkin Software. (2002). System Architect Overview - Personal Communication.

Rathwell, G. A., and Williams, T. J. (1996). Use of the Purdue Enterprise Reference Architecture and Methodology in industry (the Fluor Daniel example). In Bernus, P. and Nemes, L. (Eds.), Modelling and Methodologies for Enterprise Integration (pp. 12-44). London: Chapman & Hall.

Rumbaugh, J., Jacobson, I., and Booch, G. (1999). The Unified Modelling Language Reference Manual. Reading, MA: Addison-Wesley.

Scheer, A.-W. (1992). Architecture for Integrated Information Systems. Berlin: Springer-Verlag.

Scheer, A.-W. (1994a). Business Process Engineering - Reference Models for Industrial Enterprises. Berlin: Springer-Verlag.

Scheer, A.-W. (1994b). CIM, Computer Integrated Manufacturing - Towards the Factory of the Future. Berlin: Springer-Verlag.

Scheer, A.-W. (1998a). ARIS - House of Business Engineering. In Molina, A., Kusiak, A. and Sanchez, J. (Eds.), Handbook of Life Cycle Engineering - concepts, models and technologies. Dordrecht: Kluwer Academic Publishers.

Scheer, A.-W. (1998b). ARIS-Business Process Modeling (2^{nd} ed.). Berlin: Springer-Verlag.

Scheer, A.-W. (1999). ARIS-Business Process Frameworks (3^{rd} ed.). Berlin: Springer-Verlag.

Scheer, A.-W., and Kruse, C. (1994). ARIS-Framework and Toolset: A comprehensive business process reengineering methodology. Paper presented at the Third International Conference on Automation, Robotics and Computer Vision (ICARCV '94), Singapore.

Sowa, J. F., and Zachman, J. A. (1992). Extending and formalizing the framework for information systems architecture. IBM Systems Journal, 31(3).

Sowell, C. P. (2000). The C4ISR Architecture Framework: History, Status, and Plans for Evolution. Paper presented at the 5th International Command and Control Research and Technology Symposium, Canberra, ACT Australia.

Vail, E. (2001). Causal Architecture: Bringing the Zachman Framework to Life, [White Paper]. Ptech, Inc. Available: http://www.ptech.com [2002, Aug 2002].

Vernadat, F. (1996). Enterprise Modelling and Integration: principles and application. London: Chapman & Hall.

Vernadat, F. B. (1995). CIM business process and enterprise activity modelling. In Bernus, P. and Nemes, L. (Eds.), Modelling and Methodologies for Enterprise Integration (pp. 171-182). London: Chapman & Hall.

Vernadat, F. B. (1998). The CIMOSA Languages. In Bernus, P., Mertins, K. and Schmidt, G. (Eds.), Handbook of Information Systems (pp. 243-263). Heidelberg: Springer-Verlag Berlin.

Vernadat, F. B. (2001). UEML: Towards an Unified Modelling Language. Paper presented at the 3^e Conférence Francophone de MOdélisation et SIMulation "Conception, Analyse et Gestion des Systèmes Industriels" - MOSIM '01, Troyes, France.

Vlietstra, J. (1996). A summary of the CIMOSA reference architecture. In Bernus, P., Nemes, L. and Williams, T.J. (Eds.), Architectures for Enterprise Integration. London: Chapman & Hall.

W3C. (2000). Extensible Markup Language (XML) 1.0 (2nd edition), [Web Page]. W3C. Available: http://www.w3.org/TR/REC-xml, http://www.w3.org/XML/ [2002, June 2002].

Wagenhals, L. W., Shin, I., Kim, D., and Levis, A. H. (2000). C4ISR Architectures II: Structured Analysis Approach for architecture design. Journal of INCOSE, 3(4).

Williams, T. J. (1994a). The Purdue Enterprise Reference Architecture. Computers in Industry, 24(2-3), 141-158.

Williams, T. J. (Ed.). (1994b). A Guide to Master Planning and Implementation for Enterprise Integration Programs (Report 157). West Lafayette, Indiana: Purdue Laboratory for Applied Industrial Control.

Williams, T. J. (Ed.). (1998). CIM Reference Model Committee - International Purdue Workshop on Industrial Computer Systems. (1988) A Reference Model for Computer Integrated Manufacturing (CIM) - A Description from the Viewpoint of Industrial Automation (2nd ed.). Research Triangle Park, NC: Instrument Society of America.

Williams, T. J. and Li, H. (1995). A Specification and Statement of Requirements for GERAM (Report 159). West Lafayette, Indiana: Purdue Laboratory for Applied Industrial Control.

Williams, T. J., Li, H., Bernus, P., Uppington, G. and Nemes, L. (1998a). The Life Cycle of an Enterprise. In Molina, A., Kusiak, A. and Sanchez, J. (Eds.), Handbook of Life Cycle Engineering - Concepts, Models and Methodologies (pp. 153-179). Dordrecht: Kluwer Academic Publishers.

Williams, T. J., and Li, H. (1998b). PERA and GERAM: Enterprise Reference Architectures in Enterprise Integration. Paper presented at the International Conference on Design of Information Infrastructure Systems for Manufacturing 1998 (DIISM98), Melbourne, Australia.

Williams, T. J., Rathwell, G. A., and Li, H. (2001). A Handbook on Master Planning and Implementation (Report 160). West Lafayette, Indiana: Purdue Laboratory for Applied Industrial Control.

Williams, T. J., Zoetekouw, D., Shewchuck, J. P., Chen, D., and Li, H. (1996). Techniques to map the architectures directly against one another. In Bernus, P., Nemes, L. and Williams, T.J. (Eds.), Architectures for Enterprise Integration. London: Chapman & Hall.

Zachman, J. A. (1987). A Framework for Information Systems Architecture. IBM Systems Journal, 26(3).

Zachman, J. A. (2000a). Enterprise Architecture - A Framework, [Web Page]. ZIFA- Zachman International. Available: http://www.zifa.com [2002, June 2002].

Zachman, J. A. (2000b). Zachman Framework Definition and Enterprise Architecture Quick Start, [Web Page]. ZIFA- Zachman International. Available: http://www.zifa.com [2002, June 2002].

Zelm, M., Vernadat, F., and Kosanke, K. (1995). The CIMOSA Modelling Process. Computers in Industry, 27(2), 123-142.

Part II

Strategy Making and Business Planning

4

STRATEGY AS A CREATION OF CORPORATE FUTURE

Brane Kalpic[1], Krsto Pandza[2], and Peter Bernus[3]

[1] ETI Jt. Co. (Slovenia)
[2] University of Maribor (Slovenia)
[3] Griffith University (Australia)

4.1 Introduction

This Chapter presents an introvert perspective of strategy formation, through the presentation of the resource-based view (RBV) concept. It contains a review of a) relevant fundamentals of the strategic management field (the definition of strategy, resources and capabilities), and b) a strategy framework, which incorporates the conceptual knowledge of the RBV intertwined with some well-known tools in the domain of application.
The presented strategy framework is not intended to be a prescriptive step-by-step methodology. Instead, it offers immediate help and reminds the user of issues to be discussed in the process of the strategy formation.
The words 'strategy' and 'strategic' are well recognised and widely used (and misused) in the modern business world. The important questions to be answered are:

- whether the word strategy is used in the correct context and
- what is its real meaning?

Despite the obvious importance of strategy, there is surprisingly little agreement on what a strategy really is. However, the fact is that behind every successful company, there is a superior strategy (Markides , 1999).
Strategy is usually associated with long range planning, in a hierarchically structured system of objectives and goals, and connected to a selected way of creating a fit between external environment, internal resources and capabilities. This view is not wrong in itself, yet it is too narrow. Contemporary thoughts in the field of strategic management imply that strategy should be understood as the creation of the company's future which is the result of collective social activity, considered as an ongoing process (and not as a category) and idiosyncratic in its essence.
Despite the existence of various definitions of strategy, the temptation to offer a generic strategy definition is avoided here. Instead, various theoretical

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

traditions are presented, that populate the field of strategic management. The debate about diverse theoretical traditions is conceptualised by introducing the notions of *pragmatic* and *conceptual* knowledge.

Furthermore, the resource-based view is presented since it is one of the most dominant theoretical schools in the field of strategic management, yet often neglected by executives.

Practice shows that corporate identity and strategy are usually built around strategic business units focusing on markets and products, without sufficiently considering other essential entities of strategy, namely resources and competencies. The reason for this approach appears to be that the resource perspective is not a natural one for most companies, given that there are markets and products that characterise the immediate success or failure of a company, and a resource-based view is detached from this day-to-day reality.

While product and market position can be changed in a relatively easy way (if this new position relies on a portfolio of the same or similar capabilities that exist in the present), changes in the portfolio of resources, capabilities and competences require a longer-term evolutionary process, characterised by organisational learning.

Therefore, this Chapter emphasises a perspective of strategy where a company must be viewed not only as a portfolio of products, but also as a portfolio of resources and competencies.

In Section 4.4, a strategy framework is presented, which integrates the conceptual knowledge offered by a resource-based view with the pragmatic knowledge represented by some classical reference models and tools.

Although strategy making is in its essence a dynamic process, often not amenable for prescriptions, it is not possible to deny entirely that management as an applied discipline demands normative solutions that seek to guide managers' activities.

The authors are aware of the difficulties concerning- and incompleteness in the design of any normative prescription (for instance in the form of formal step-by-step reference models). However, because of the nature and purpose of this Chapter, some practical guidelines and abstract models have to be offered in order to support the process of managing strategy life-cycle and understanding strategy itself.

Thus, the strategy framework presented here should help directly and remind the user of issues to be discussed in the process of strategy formation and actions to be taken in its implementation.

4.2 What is Strategy and Why it Matters?

A historical perspective reveals the military roots of the word strategy. In ancient Greece, the word 'strategos' has been referred to as the role of an individual – a general in command of an army. Strategy has been represented

as an art in general and as a set of behavioural and psychological skill of an individual.

Pericles (450 B.C.) linked the word 'strategy' with managerial skills for the first time in history, and Alexander the Great (in 330 B.C.) defined strategy as a skill of employing forces to overcome opposition and to create a unified system of global governance.

Academic underpinnings of the field of strategic management can be traced back to the beginnings of the twentieth century. Beginning in 1912, Harvard offered a course in 'Business policy', which was designed to integrate knowledge gained in functional areas like accounting, operations, and finance, thereby giving management students a broader perspective on the strategic problems faced by corporate executives. Strategy can therefore be understood as a metaphor that spans the functional areas in business. What is more, strategy making is a genuine interdisciplinary field involving economics, management, organisational theory and law.

This interdisciplinary nature of strategy making and the multi-faceted phenomena that influence strategic behaviour of a company increase the ambiguity of any strategy. Despite nearly a century of teaching of the subject and academic research, there is still considerable disagreement about precisely what strategy is, and especially how to develop a good strategy. Although lack of widely acceptable definitions and non-existence of generic prescription might question the legitimacy of the field, the authors believe that it is more likely that a better understanding of the complex world of business is to be gained from different perspectives and theoretical traditions.

Schendel (1994) argues that because strategic management is fundamentally an interdisciplinary subject, a field of practice and application, where the perspective will shift and research approaches will be incommensurable, it is unlikely that a single paradigm will ever govern the field. Using words more appropriate to a practicing manager, it is unlikely that one will ever be in the position to offer a generic prescription of how to develop a good strategy or to create theories that can precisely explain why a certain company is more successful than the other. In other words, no advice will apply to all firms all the time. What a firm should do depends on its own particular circumstances, which are in turn determined by its stage of evolution. Strategic advice that fails to put the company in its historical context runs the risk of putting a company into danger.

However, a strategy should by no means be interpreted as a metaphysical category. Markides (1999) argues that behind every successful company, there is superior strategy. A company may have developed its strategy through formal analysis, trial and error and experimenting, intuition or even pure luck. No matter how it was developed, it is necessary to understand the logic of successful strategies. If the building blocks of successful strategy are understood, it is more likely to develop a new strategy when the current one runs its course. This leads to the conclusion that managers need knowledge that helps them make sense of a complex business reality rather than plug-and-play

techniques or a 'user-manual' type of knowledge so often offered by consultants. Understanding the external business context and internal endowments with resources and capabilities creates a corporate future.
In order to present various perspectives on strategy, different authors use different classifications. Mintzberg (1998) operates with ten popular, major schools of thoughts. Whittington (1993) offers a typology of four generic perspectives on strategies, namely classical, evolutionary, systemic and processual. Ghevamat (1999) identifies four basic stages that involve grappling with increasing levels of dynamism, multidimensionality and uncertainty and that therefore become less amenable to routine quantitative analysis. From a historical perspective, strategic thoughts have developed from financial planning and forecast-based planning to externally oriented planning and to the present state labelled 'strategic management'. This latter changes the focus from *predicting*, to *creating* the future.
In this Chapter, the debate about different schools of thoughts in the field of strategy is conceptualised around two different types of knowledge that populate the field of management in general and strategy in particular. *Pragmatic* knowledge and *conceptual* knowledge present two categories used to frame the discussion.

4.2.1 Pragmatic Knowledge

Pragmatism advocates knowledge that facilitates problem solving. It is deemed that most practicing managers claim to be pragmatist. They are more likely to focus on what works than on conceptual knowledge. This philosophy implies an emphasis on practical knowledge that is explicitly concerned with how to get things done. No matter how sophisticated and complex a tool is, it is just a tool that enables and facilitates a certain work or activity.
Pragmatic knowledge is usually represented in different utilitarian (practical) frameworks, which are focused on the direct use of this knowledge to managers. These frameworks are usually prescriptive and deterministic. Although phenomena of strategic interest are usually wide in scope and limited in detail, these frameworks still claim to offer generic solutions.
Pragmatic approaches to strategy making assume that companies can adapt to environmental changes by restructuring themselves in an intentional manner, making educated or informed strategic choices. This perspective assumes that managers have the ability to intervene and successfully influence the direction of organisations. Since senior management as a rule drives strategic direction, managers want to be equipped by utilisation frameworks that enable them to execute the strategy formation process.
In the strategic management literature pragmatic knowledge is prevalent in the classical perspective (Whittington , 1993). Within this perspective, strategy making is a rational process of deliberate calculations and analysis designed to maximise profit. The main tenets of this school of thought are that the capacity of 'the strategist' is to provide explanation internally by analysis

of phenomena present in the environment, and to forecast these phenomena deliberately and proactively into the future.

This top-down, planned and rational approach to strategy making emphasises the explicit and deliberate conception of goals, and the logical cascading of actions and resources derived from original objectives. Mintzberg (1998) refers to this perspective when discussing the 'design school', 'planning school' and 'positioning school'. These three schools are labelled as 'prescriptive' since they offer 'instructions' on how to develop strategy.

According to Mintzberg, the *design school* presents the original perspective where a strategy is a deliberate process of conscious thought. The *planning school* has grown in parallel with the design school. The planning school has accepted most of the premises of the design school, however, it did alter one premise, and this has made all the difference. Strategy formation should be a controlled, conscious, and formal process, decomposable into distinct steps, delineated by checklists, and supported by techniques. The basic notion supported by this school advocates the importance of formal planning, yet surprisingly little empirical research was conducted to explore how planning really effects performance.

The *positioning school* pioneered by Porter (1980) holds a special place among schools that nurture pragmatic knowledge. It has altered the classical orientation towards the process of strategy formulation towards emphasis on strategic content. By doing this, the positioning school has made a step forward towards conceptual knowledge. The *Five Forces Framework* of Porter was the dominant perspective at least during the 1980s. In Porter's Framework, the five forces - entry barriers, threat of substitution, bargaining power of suppliers and buyers, and rivalry among industrial incumbents - determine the inherent profit potential of an industry. The knowledge accumulated by this school can be used to help the firm find a product/market position in an industry from which it can best defend itself against competitive forces or influence them in its favour.

The reasons why this school is classified among the pragmatic schools are threefold. First, it accepts most of the premises that underlay the planning and design school. Second, it suggests a very straightforward decision-making process:

- Pick an industry based on its structural attractiveness;
- Choose an 'entry strategy' based on conjectures about competitors' rational strategies;
- If not already possessed, acquire or otherwise obtain necessary resources to compete on markets.

Third, the positioning school is often reduced on choosing a generic strategy such as cost leadership, differentiation, and focus. This generic strategy framework introduced the need to choose a position in order not to become caught up in-between. Trade-offs between generic positions are essential to strategy. They create the need for choice and purposefully limit what a company offers.

At this point however, it has to be mentioned that in his more recent work, Porter (1996) slightly departs from generic strategy by arguing that a company should find a unique position that asks to perform activities differently from rivals.
All theoretical traditions that nurture pragmatic knowledge prove to be very lucrative for consultants. Various portfolio analysis such as the Boston Consulting Group's 'Growth-Share Matrix', or McKinsey's 'The Industry Attractiveness – Business Strength Matrix' have become core tools that should enable the formulation of strategy in an analytical-planning mode. Ghevamat (1999) refers to these frameworks as 'powerful oversimplifications' that claim to offer generic remedies for solving problems related to strategy formulation and implementation.

4.2.2 From Pragmatic to Conceptual Knowledge

Fundamental tenets of the traditional rational-planning mode of strategy have been seriously criticized by the academic community, although they still remain influential in the consulting society selling universal tools for strategy making. Practicing managers largely appreciate such prescriptive tools when formulating strategy, although this often leads to a vast plan that instead of guiding the company, formulates what is already considered common sense knowledge.
The fact that strategy formation is recognised as a cognitive process is a difficulty when strategy making is attempted to be understood as a rational plan. Cyert and March (1963) introduced '*bounded rationality*', which implied that the 'rational economic man' is a fiction. Assumptions about rational choices are described as inadequate foundations for understanding strategy. Lane et al. (1996) address these inadequate assertions. Bounded rationality implies that not every strategic action is a result of choice. What is more, a strategic choice often occurs when an event or action already took place. The cognitive limitations of the human brain do not allow us to identify a complete set of consequences that describe what might happen if a decision is made. It is often impossible to make a decision based on the predicted consequences associated with it.
The cognitive limitation challenges the image that strategy is a course of action consciously deliberated by top management or an analytical exercise undertaken by staff strategists. This has led some strategic management scholars to describe how strategy is actually formed instead of prescribing what it should be. Findings from their empirical studies suggest that strategy is, more or less, emergent from lower levels of organisations (Mintzberg , 1978; Mintzberg and Waters , 1985), whether through trial-and-error, learning, and or emerging incrementally with logical guidance from the top.
Mintzberg 's (1978) impressive study of Volkswagen strategies from 1920 to 1974 shows that strategy is a pattern in a stream of decisions and actions, where intended and emergent strategies intertwine. In the conclusion of that

research, it is argued that a strategy is not a fixed plan, nor does it change systematically at pre-arranged times solely at the will of management.
Such studies have gradually eroded the popularity of different problem-solving frameworks and opened the space for generating more conceptual knowledge. The strategic management field has established its identity around a few theoretical questions (Schendel , 1994):

- Why do firms differ?
- How do firms behave?
- What explains the success and failure of firms?
- How does the strategy making process affect strategic outcomes?
- How do differences persist?

These questions are vital for the understanding of the strategy of a company. To find answers, it is necessary to generate conceptual knowledge that goes beyond short-term problem solving.

4.2.3 Conceptual Knowledge

Explaining and understanding why firms succeed or failed, why they differ, how they behave, how they choose strategies, and how they are managed, requires *conceptual* knowledge. This kind of knowledge represents a shift from the search for prescriptions as to how to get things done to a more loosely defined activity of sense making and creating understanding. The strength of conceptual knowledge, when its implications for managers are considered, does not rely on its ability to prescribe managerial actions. With its concern for creating understanding and making sense of a complex organisational reality, this knowledge can provide a touchstone for managers when confronting an ambiguous situation. Conceptual knowledge is valuable when it confronts practitioners with an idea which cannot simply be ignored.
Theoretical traditions that nurture conceptual knowledge alter the basic premises of the strategic choice perspective. A strategy can be conceived as the emergence of collective social action. Whereas the strategic choice view the managers as making decisions to maximise some long-term advantage, a deliberate plan that can be successfully carried out, the alternative approach takes a view of the world in which firms create differential behaviour which maps into advantages, but these advantages cannot be 'planned for'.
The theoretical tradition most sceptical about top management's ability to plan and act rationally is a form of determinism. A deterministic view of strategic management actually implies that management has a passive role and is largely unable to influence change and long term survival. This school can be found in diverse theories, from the application of Darwinian theory in the organisational context such as structural inertia theory (as described in Hannan and Freeman (1984)) and transaction cost economics (Coase , 1937; Williamson , 1985) developed within industrial organisational economics literature, to the application of chaos theories (Brown and Eisenhardt , 1998).

The advantage of the *'Darwinian'* school of thought is that it links change in the external environment to the internal environment. However, it reduces the role of managers to passive observers and ignores learning from prior strategies.

Transaction cost economics (Williamson , 1985) claims that the only comparable advantage is efficiency. Managers must concentrate on their cost, especially the transaction costs of organising and co-ordinating. The main purpose of transaction costs economics is to explain why economic activity is organised as it is within firms. Firms and markets are seen as two alternative means of co-ordination, the firm being characterised by co-ordination through authority relations and the market being characterised by co-ordination through price mechanisms. The conceptual knowledge offered by transaction cost economics is essential for understanding make-or-buy decisions, mergers, acquisitions and strategic alliances.

The *'Chaos'* school addresses the complexity and unpredictability of management. In contrast to structural inertia theory it rejects ecological equilibrium. As the external environment is chaotic and unpredictable, so the idealised hierarchical strategy formulation process is also modified by spontaneous and chaotic elements. The main advantage of the application of chaos theory is to address the issues of unpredictable, intense and high-velocity changes characteristic for modern industry. This school focuses on internal response to chaos such as time pacing, semi-coherent strategic direction, continuous flow of competitive advantage and organisational culture. The main message embedded in chaos theory's application is to remain flexible in an uncertain world. This asks for Value Strategies as real options.

Shapiro 's article (1989) titled 'The Theory of Business Strategy', announced the emergence of a new approach to strategy. This approach utilizes the tools of *game theory* to analyse the nature of competitive interaction between rival firms[4]. The main thrust of work in this tradition is to reveal how a firm can influence the behaviour and actions of rival firms and thus the market environment. Game theory attempts to develop a better understanding of a firm's environment by viewing business as a series of moves and countermoves. The premise of game theory is that, to identify your best moves, you must first analyse your opponent's most likely countermoves. This requires seeing the world through your competitor's eyes as well as through your own. Game theory provides a set of analytical tools, which can help us understand the phenomena we might observe when decision makers interact. Game theory is being used by a growing number of companies to make decisions about marketing variables, capacity expansion and reduction, entry and entry-deterrence, acquisitions, bidding, and negotiation.

The theoretical tradition that has had a tremendous influence on strategic management research during the past two decades is called the *resource-based*

[4] For an introduction to game theory and its use in strategy making refer (Brandenburger and Nalebuff , 1995).

view of the firm. This theory's explanation for the persistence of competitive advantage and its sources has received the greatest attention. This perspective focuses on the internal organisation of firms, and so is complementing the traditional emphasis on industry structure and strategic positioning.
The resource-based view assumes that firms can be conceptualised as bundles of resources and capabilities. Note that the capability perspective slightly differs from the traditional resource-based view. The resources and capabilities with which firms compete cannot be bought or sold in markets. Especially capabilities must be developed rather than being taken as given. In other words, the main message of the resource-based view is that the sources of sustainable superior performance lie internally, in the capacity to exploit and deploy resources, rather than externally, in simply positioning the firm in the right markets.
At this point, where different theoretical traditions have been presented, it is important to say that these perspectives are in many ways complementary and a full understanding of the word strategy requires an appreciation of all approaches. Especially, because managers have to address multiple aims and understand often-ambiguous internal and external aspects of competition that can only be explained from the viewpoint of several perspectives of strategy. Facing the need for a comprehensive understanding, managers have to integrate multiple perspectives. By doing this they balance between rational analytical planning issues, emergent issues, deterministic evolutionary forces, developmental issues, and learning.
To summarise thoughts presented in this section, it is important to say that although there is difficult to find a generic definition about strategy the latter is very important. The strategic direction of the business organisation is at the heart of wealth creation in modern industrial society. Understanding how firms develop, maintain and advance their organisational knowledge is fundamental to understanding how our society works and how it changes.

4.3 Resources and Capabilities

4.3.1 An Introvert Approach to Strategy Creation

The resource-based view (RBV) of the firm and the dynamic capabilities approach (DCA) constitute two distinct, yet highly related streams of research in strategic management literature. Both streams have been proposed to understand intrinsic firm heterogeneity that accounts for durable differences in performance. Makadok (2001) refers to this difference when discussing distinctions between resource-picking and capability-building mechanisms. From the RBV perspective the heterogeneity in performance is due to the ownership of different resource bundles with differential productivity (Amit , 1993; Barney , 1986; Conner , 1991; Mahoney and Pandian , 1992; Wernerfelt , 1984). DCA represents a shift from observing static asymmetries in resource endowments

among firms to a more dynamic exercise of deployment and redeployment of resources. The focus of DCA is on how firms develop firm-specific capabilities that allow them to respond to shifts in the business environment (Eisenhart and Martin , 2000; Nelson and Winter , 1982; Teece et al. , 1997; Winter , 2000). DCA altered the focus from considering the firm-specific resource structure to exploring organisational processes, routines, learning and knowledge.

Despite the fact that the RBV and DCA literature is conceptually very comprehensive, little effort has been made to develop the practical implications of these schools for managers. This is not necessarily due to the unwillingness of researchers to provide prescriptive frameworks. It is more likely that the avowed complexity of organisational phenomena, such as capability development, makes it difficult for them to develop prescriptive approaches and frameworks. The complexity of capability development and its evolutionary nature largely constrain managerial actions and make both approaches less suitable for normative prescriptions. Attempts to offer normative implications will have to confront several impediments to such endeavours.

4.3.2 Impediments for Normative Prescription

Strategy scholars are constantly challenged to prescribe how to achieve competitive advantage. However, it may be logically impossible to formulate a set of rules to systematically create competitive advantage. RBV and DCA present two approaches that explicitly explain why competitive advantage is achieved or even sustained, but they remain less powerful in prescribing how to manage the development of resources and capabilities.Furthermore, the intrinsic logic of both approaches that emphasise complexity, path-dependency, idiosyncrasy of the phenomenon in return creates impediments for normative prescriptions. Every framework with an ambition to guide the development of capabilities will have to confront these properties. These impediments may be classified into two categories: system complexity and process complexity.

4.3.2.1 System Complexity

System complexity can be introduced by discussing ambiguity of available definitions about resources and capabilities. Ever since Wernerfelt (1984) defined resources as anything which could be thought as a strength or weakness of a given firm, attempts have been made to provide concrete definitions. It was deemed that clear definitions would help make RBV more operational.

In general a complex system consists of a large number of individual agents that can change their behaviour on the basis of information they receive about what the other agents in the system are doing.

A *resource* is usually defined as a firm's *asset*, which implies that it is owned. Resources can be *tangible assets* such as facilities and process technology or *intangible* such as patents, brand name, reputation and trade secrets (Hall ,

1992). Additional resources can be classified according to the extent to which they are firm-specific. Teece et al. (1997) has defined resources as *firm-specific assets* such as specialised production facilities or engineering expertise. This implies that ordinary, *non-specific assets* do not represent firm resources. However, the Wal-Mart case documented by Stalk et al. (1992) shows that non-specific assets such as real estate, trucking fleet and information technology - when productively linked together - can constitute a powerful logistic capability.

If a resource is understood as a more or less firm-specific asset to which a monetary value can be attached, a *capability* refers to a firm's capacity to deploy resources. Collis (1994) defines capability as a firm's capacity to more efficiently or effectively choose and implement the activities necessary to produce and deliver a product or service to customers. More recently, Eisenhart and Martin (2000) have defined capability as a set of specific yet identifiable processes that are idiosyncratic in their details and path dependent in their emergence. Makadok (2001) refers to capability as to a special type of resource whose purpose is to improve the productivity of the other resources possessed by the firm.

Examples of capabilities are often discussed by categorising capabilities into different levels. Verona (1999) has classified capabilities into *functional* and *integrative* capabilities. The former allows a firm to deepen its functional knowledge such as scientific expertise, manufacturing knowledge and marketing expertise. The letter combines different functional capabilities and additionally absorbs critical knowledge from external sources. It may prove helpful to add *cross-functional* capabilities. They represent identifiable processes characterised by their multi-functional nature such as: product development, quality management, and flexibility in meeting customer demands, and speed and dependability of order fulfilment.

Although the literature already offers enough definitions to make sense of what is a capability and a resource, this does not necessary mean that these help to manage them. Even though the definitions would allow to identify every single resource and disaggregate every capability, they fail to explain the relationship among them. Brand name and reputation are more likely resources that are outcomes of a network of functional, cross-functional and integrative capabilities, than resources which underpin marketing capabilities. On the other hand, a firm-specific advanced process technology developed in house may be an outcome of R&D and manufacturing expertise. Such a resource in return supports a basic manufacturing capability and different cross-functional capabilities, such as fast new product development or flexibility in response to customers' demands.

An integrative capability can refer to a firm's capability to use external resources productively. Gulati (1998) defines network resources as entities in networks that provide informational advantage. Through networks, firms can obtain access to resources that create value and to capabilities that require time to build up. It means that from a single firm's perspective a capabil-

ity can be seen from the network's perspective as a resource. What is more, a firm's network which can be seen as an idiosyncratic resource is created through a path-dependent process and is therefore, due to its characteristics, more akin to a capability than to a static asset. Organisational routines and organisational culture may conceptually be seen as integrative capabilities. However, it does not necessary mean that they just co-ordinate different functional capabilities and integrate them into cross-functional capabilities; the aggregate capability is a capability in its own right.

This leads back to the genuine idea of RBV, namely that a firm can be seen as a system of capabilities and resources. Black and Boal (1994) define a system resource, which is created by a complex network of the firm's resources. The notion of a system resource implies that a capability is a socially complex system. Capabilities and resources are not singular and distinctive items. They do not exist in isolation from each other. In other words, they are meaningful and valuable just if they are linked. The complexity of such a system is caused by an extreme interdependency among the constituent elements.

System complexity has been embedded in RBV from its beginnings. Heterogeneity implies that resources have different productive value depending on how they are linked with a firm-specific structure of resources. Conner (1991) argues that the productive value of a resource is higher if it is more specific to the system of other resources. This productive value - achieved in connection with other resources - and capabilities has to be bigger than the value of a single resource on the market (Barney , 1986). Barney argued that this market for resources is imperfect since firms have different expectations about the future value of strategic assets. Didrickx and Cool (1989) expressed criticism about the neoclassical assumption that a market with strategic resources exists. They claimed that strategic resources are non-tradable on the market and they are more likely to be accumulated than acquired. This implies that "opportunity" costs cannot be assessed and therefore the difference between the resource's value when connected with a system of firm-specific resources and the value of its alternative use is less relevant. This criticism shifted focus of a debate from value and productivity to the characteristics of resources. It was deemed that resources and capabilities which are *valuable, non-tradable, rare, inimitable* and *non-substitutable* can present a source of *competitive advantage* (Barney , 1991; Didrickx and Cool , 1989).

The debate about characteristics of resources and capabilities has not improved the practical implication of RBV and DCA. In this debate, strategic resources and capabilities have become almost mystical categories. They became tacit, ambiguous, non-transparent and therefore almost unmanageable. It is hard to imagine how to manage and sustain something what is described as causal ambiguous, which implies that nobody, except maybe the entire firm itself can describe the origins of its competitiveness.

What asks for more attention is the system nature of resources and capabilities. They are identifiable categories, however, when linked together it is difficult to assess the value (extent) of an individual capability or the produc-

tivity of a single resource. For a manager it is difficult to develop a causal understanding of all relations among capabilities and resources, which is a problem, because such understanding would in turn enable a far more successful management. Management of capabilities and resources is a holistic activity since managing a single capability means that one has to manage all other capabilities and resources connected to it. However, it is difficult to assess how an investment in a resource or a project with the aim of developing a capability will affect the productivity of other linked resources and the value of other capabilities. This issue can also be viewed from reversible perspective. Every investment, project or initiative that is addressed mainly to create or change a single resource or a capability, creates an option for other resources to be more productive and other capabilities to be more valuable. What is more, every such investment or project provides a firm with options to respond to different market opportunities.

4.3.2.2 Process Complexity

System complexity arises from a high level of interdependency among capabilities and resources. DCA scholars argue that *capability development is a highly dynamic phenomenon.* Such a phenomenon cannot be adequately explained through a cross-sectional perspective. It may prove to be challenging for researchers to adopt an evolutionary perspective and build process theories. This evolutionary nature of the phenomenon reveals the constraints on management action: it is the process of capability development that can be influenced through management action, while rational planning of capabilities is limited. For example, Fujimoto (2000) adopted this evolutionary perspective challenging the stereotypical notion that Toyota is a monolithic organization where changes are made by one-shot rational decision-making. From this it follows that management needs to do two things:

- create and direct processes that lead to new capability development, and
- recognise and nurture outcomes that can be distinguished as new capabilities that emerge from the above process.

Dosi et al. (2000) argue that a capability represents organisational knowledge (i.e., knowledge that the organisation has, which is more then the sum of knowledge of all individuals in the organisation). It is assumed that firms know how to do things. This also implies that managers are part of this phenomenon, not the independent agents that control and guide the process of capability development in a rational planning mode.

The process complexity of capability development results *uncertainty* and *path-dependency.* Capability development is a form of *long-term organisational learning.* Miner et al. (2001) characterise long-term learning as an improvisational, experimental and often trial-and-error process, thus definitely not entirely deliberate. Emergent and chance events decisively shape this process.

This long-term learning is stochastic by nature, since the outcome of the learning process cannot be ex-ante rationalised. Uncertainty means that managers are engaged with the process in which knowledge is accumulated, maintained, acquired, extended and potentially even lost - yet, not being in the position to accurately monitor and guide the progress of the process.

Path-dependency means that decisions made in the process are irreversible. Path-dependency is a significant impediment for managers. While decisions are taken under uncertain conditions, the results of past decisions are constraining the decisions that can be taken in the future.

The system complexity and process complexity are linked. They mutually influence each other's level of complexity. Winter (2000) argues that capabilities emerge in primitive forms. This implies that system complexity is low in the initial phase of capability development. At this point managers confront a causal ambiguity since they do not understand in which way the process is going to progress. They do not know which resources and capabilities will be needed and how they should be connected. In the process of capability development system complexity is increasing. By the time a capability reaches the status of a source of competitive advantage the system complexity rose. The causal ambiguity of initial phase is gradually transformed into causal understanding. System complexity also influences process complexity. When causal understanding about a capability is developed, mangers influence the process by decisions aimed at maintaining and upgrading the existent capability.

The dynamics of process complexity and system complexity challenge another assumption about capability. A capability is often understood as something always valuable. In other words, a firm possesses a capability or it does not. Winter (2000) refers to this binary understanding of capability when arguing that a capability is multidimensional and therefore in different dimensions its value may be different. This leads to another dynamic characteristic of a capability. Its value in business environment is changing. Every firm possesses an organisational knowledge. However, this knowledge is differently valuable in different time periods and in different business contexts. The value of this knowledge can be reduced if some significant capabilities or resources that constitutes a system are destroyed or more likely it is surpassed by more valuable capabilities developed by competitors or made obsolete by dynamic changes in the market.

Considering system complexity and the dynamic nature of the phenomenon it is deemed that capability development is not something that is easily controlled with normative and prescriptive frameworks. Helpful frameworks for managing the capability development process have to provide managers with understanding. They have to help managers to make sense when they confront the uncertainty of this phenomenon. Managers need to be able to make sense of the value of organisational knowledge. They have to be able to identify signals from the market, because it is these market signals that determine the value of possessed capabilities.

4.3.3 A Simple Abstract Model of Capabilities

Modern strategic management in the business environment attempts to concentrate on a theme, which is characterised by attention to resources, capabilities, core competencies, core products, and competitive advantage.
In spite of comprehensive literature about strategic management, the reader can hardly find one single, accepted definition or a simple abstract model, which describes and explains these basic notions and their relationships.
Although the authors are aware of the incompleteness of the presented abstract model and its limited consistency with existing definitions, they believe that this abstract model serves the purpose to present the basic relationships between business processes, resources and capabilities, in a simple and understandable way.
Organisations incorporate many interrelated business processes and activities to deliver a final product (goods or services) to the market. 4.1 presents an abstract model of the enterprise composed of business processes and activities delivering different types of outputs. These outputs could be:

- inputs to subsequent processes (the output of business processes BP1 is the input of business process BP3 – e.g., technical documents generated in the design process of a new product are the input to product technology development),
- resources for subsequent processes (the output of business processes BP2 is a resource necessary to carry out business process BP3 – e.g., the development of a new technology or specific assembly device is a resource for a manufacturing process), or
- final products delivered to the market.

Business processes and activities could employ (for their execution):

- people (characterised by their knowledge, experience, skills and talents),
- machines, devices and tools (characterised by their technical characteristics and constraints),
- methodologies, systems or tools installed in the organisation, and/or
- various types of tangible assets (e.g. buildings, real estate, etc.) and intangible assets (patents, brand names, etc.).

All of the above may be owned by the organisation and/or hired from outside of the organisation. Therefore, a business process integrates a set of resources (where any particular resource could be described by a set of attributes to define its characteristics), and must meet certain predefined criteria or requirements for activity/process performance.
According to the definition in Section 4.3.2, resources / assets could be divided into assets non-specific to the company (general purpose assets which could be acquired on the market, like machines, computers, software, etc.) and company-specific assets (which are rare, non-tradable or inimitable assets).

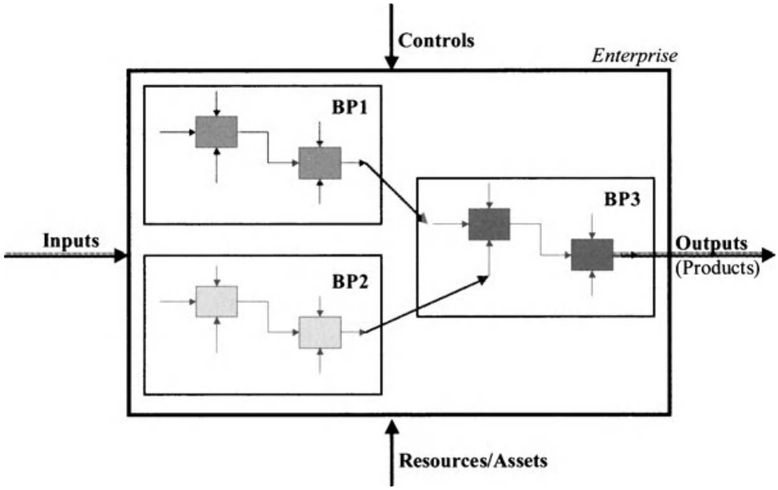


Fig. 4.1. Enterprise reference model (BP1,2 and 3 denote busines processes)

While the governance of each individual company-specific assets is of strategic importance (because of the unique nature of those assets), the management of non-specific assets needs to concentrate on how these assets have to be linked and clustered together into the cohesive whole, and if the way of linking these assets is specific - and therefore of strategic importance (see the case of Wal-Mart).

Definition: *Capability*

Capability as a basic notion means a *firm's ability to execute business processes* and activities to produce and *deliver a required product* through the deployment of the firm's resources. Therefore, a capability is a *permanent or temporary aggregation of non-specific and/or specific assets* needed to execute certain business processes.

A company may possess different types of capabilities, such as:

- functional capabilities,
- cross-functional capabilities, and
- integrative capabilities (governance capability).

According to ISO 9000:2000, business processes can be divided into a group of key business processes (or customer oriented and related processes) and support business processes (e.g. the accounting process in a manufacturing company). In general, a company could incorporate all required capabilities (of different categories) to successfully perform both groups of processes (e.g., new product development and accounting).

Definition: *Core Competence*

Capabilities of strategic importance (functional, cross functional or integrative capabilities linked and used to produce the firm's core-products),

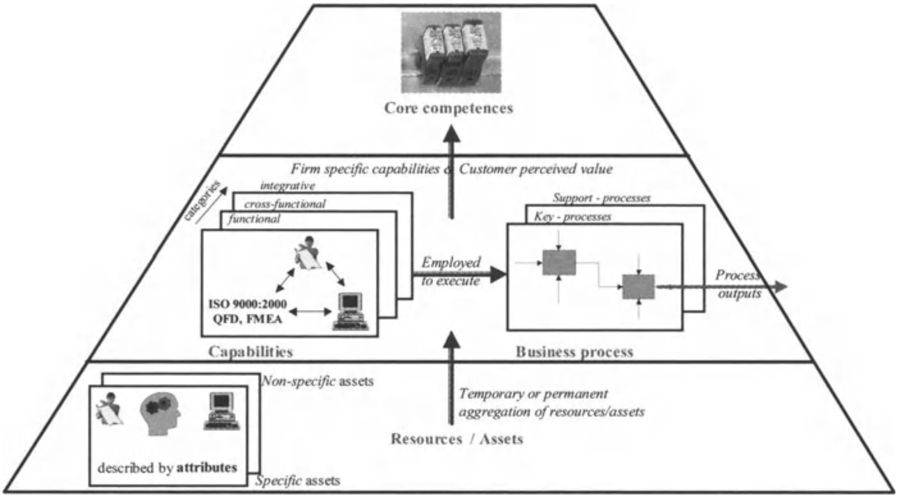


Fig. 4.2. Resource/capability abstract model

which directly contribute and improve the value perceived by the market and customers are core competencies of the organization (Prahalad and Hamel , 1990).

Take, for example, a manufacturing company that has developed its own, very efficient, accounting system supported by a company-specific software. Even though this is a specific asset of the organisation because it does not directly contribute to the value of products or services perceived by the customer, the accounting capabilites can not be considered as a core competence.

A core competence is a company-specific capability (of strategic importance), which makes the company distinct from its competitors, and defines the essence of the company's business.

Firm-specific (core) capabilities may also be considered through the perspective of the firm's *competitive advantage.* Namely, governance over core capabilities should result competitive advantage for the firm. To sustain competitive advantage for the company, core capabilities and resources should be identified, developed, accumulated and maintained through a long-term evolutionally process and organisational learning.

Therefore, the acquisition of general and tradable assets is unable to create competitive advantage - simply because those assets are also available to other competitors. Such assets may quantitatively improve the company's competitiveness, but their possession alone can never create competitive advantage.

4.3.4 Core Competencies and Core Products

Extensive research of different industries has shown a tangible link between identified core competencies and the end-products of companies (refer Fig.

4.3). This link is what is called a *core product* and is the *physical embodiment of one or more core competencies.*

Canon, for example, produces a wide and diversified portfolio of end products from binoculars to calculators, cameras, camcorders, copiers, digital camera products, document management systems, faxes, medical products, picture management, printers, scanners, security lenses, TV products, Visual Communication Systems, etc. At the first sight this seems to be an extremely wide portfolio of unrelated businesses in terms of customers, distribution channels, etc.

However, from the perspective of underlying core competencies, the company is seen as conducting a very focused and coherent business. Canon's wide portfolio of end-products relies on a much smaller set of core competencies, namely in the area of optics, microprocessor control and imaging (refer Fig. 4.3).

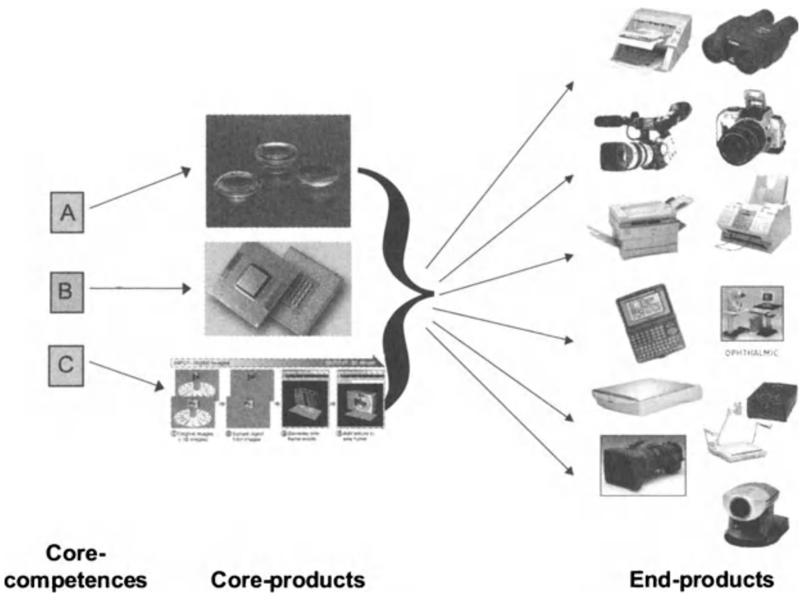


Fig. 4.3. Core competence embodiment – the Canon case study

Control over core products is essential because this allows a company to shape the evolution and different applications of its competencies through embodying them in end-products. Well-targeted core products can also lead to economies of scale or scope, and sometimes to a dominant position over competitors.

The Canon case also shows that the portfolio of core competencies may be narrow - no matter what the size of the company. However, it should be noted that concentrating on a too narrow set of core competencies has an element

of risk for the company. Namely, the competitive environment may produce competitors who duplicate this small set of competencies and the strategic (or competitive) advantage of the company could easily disappear.
Unfortunately, companies often have difficulties recognising the source of their competitive advantage. They can hardly identify their core-capacities and products, and in many cases, management still perceives the company as a bundle of businesses making products, and not as a portfolio of accumulated capabilities and company-specific resources.

4.3.4.1 A Capability Identification Case Study

The identification and understanding of capabilities and resources is a foundation for strategy making, as demonstrated in the example below (refer Fig. 4.4):
A company incorporates several related strategic business units (SBU) where the two largest SBUs represent the production of fuses and the production of circuit breakers (RCB).
The company intuitively gave a priority (in its R&D budget, marketing activities, etc.) to the SBU of circuit breakers. This priority was justified by the fact that the RCB market is larger and faster growing than the fuse market (which is therefore supposed to be phased out on the long run).
However, detailed research and matching of required and existing capabilities has shown a picture contrary to what was expected. For retaining its competitive position in the RCB market, the company could develop at least one of the following competencies:

- R&D excellence in the development of new products resulting in technically advanced products or highly integrated systems (complete solutions for a particular market segment), or
- Operational excellence manifested in the responsiveness to customer demands and reliability of incorporated processes (functional and integrative capabilities).

However, none of the mentioned capabilities could be considered a source of potential competitive advantage. In addition, the research process couldn't identify the core product either.
The situation was quite different in the review of capabilities of fuse SBU where the focus and scope of core competencies has been defined in the following way: 'The development and production of any kind of fuses (form commodity type of products to special types) on the ceramic base'.
To improve its competitive position, the company could:

a) integrate and incorporate some of its existing capabilities, which include:
   - its long accumulation of knowledge necessary for the development and production of ceramic bases (fuse bodies) in contrast with plastic casing and simple metal components of RCB products which could be also produced in small workshops,

- its capability to develop and produce special tools for fuse component production,
- its capability to develop and build special assembly lines for the complete automation of fuse assembly,
- the capability for highly automated production of fuses;

and

b) improve existing and acquire new capabilities for:
- R&D excellence in the development of any type of fuse for achieving customised design (in accordance with all major world-wide standards),
- development and production of special types of fuses which require highly cooperative work of specialists from the field of ceramic (ceramic bodies) and end product development (fuses with high built-in engineering content),
- marketing of some special families of fuses,
- to be able to create a powerful core competence and dominant position in fuse production.

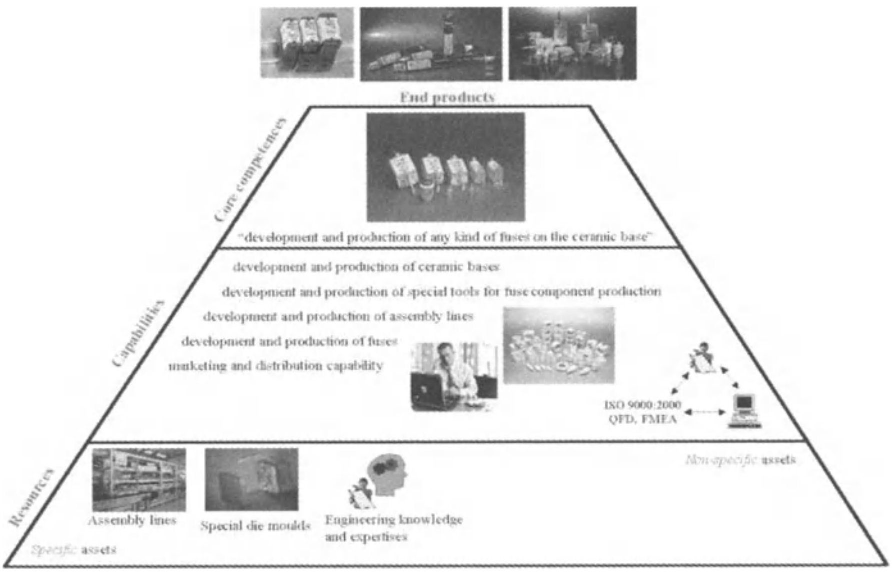


Fig. 4.4. Portfolio of competencies

The company also has an explicitly defended core product – ceramic components of fuses. It has to be emphasised that the company's competitors usually do not possess their own manufacturing resources and capabilities for ceramic components (in some cases the company supplies competitors with

ceramic components). Therefore the segment of special fuses which rely on specific ceramic components, produced individually or in small batches, with highly incorporated engineering content and cooperative work of ceramic and fuse experts in new product development, represents a potentially powerful capability which can create a distinctive advantage for the company.
Beside the production of the core product for its own production purpose, as previously mentioned the company supplies many competitors with ceramic bases, hence holding a dominant position over competitors.
Regarding the accumulation of capabilities in the development and manufacturing of ceramic and electrotechnical products, the company should find a new product or market space where it could apply its acquired capabilities (capability stretch and leverage).
To improve its competitive position the company also has to improve some cross-functional capabilities (flexibility and quality), and integrative capabilities (using the concept of outsourcing).

4.4 The Strategy Process

The research of strategic management practice in the business environment shows examples of brilliant strategy crafting and implementation, but also many evidences of failures.
Why is it that companies have so many difficulties with strategy formation? Strategy formation takes a substantial and sustained intellectual energy to develop high-quality, robust answers to questions such as:

- what new competencies will need to be build?
- what new product concepts should the company pioneer?
- what new alliances does the company need to form?
- what economic activities must be kept in-house?

Companies often perceive the strategy formation as a one-time exercise undertaken by company executives, who by employment of analytical tools create some basic figures and projections about the company's revenues, market share, etc. These figures are then written up and later presented in the strategic plan of the company. Strategy and strategic plan are not regularly reviewed and they can remain unchanged to the end of the time horizon taken in the strategic plan.
Many times strategic documents show the lack or the incompleteness of important answers, which could give a strategic direction to the company, and ensure consistency of decisions making leading to the long-term development of the company's capabilities and resources. Therefore strategy making cannot be considered as a one-time effort but it must be an ongoing process which should provide a continuous checking of the firm's course toward its strategic objectives and its vision.

Some typical questions asked by managers at this point are: how do we create an efficient strategy?, through what steps should we proceed?, what are required 'ingredients' of the strategy?, etc.

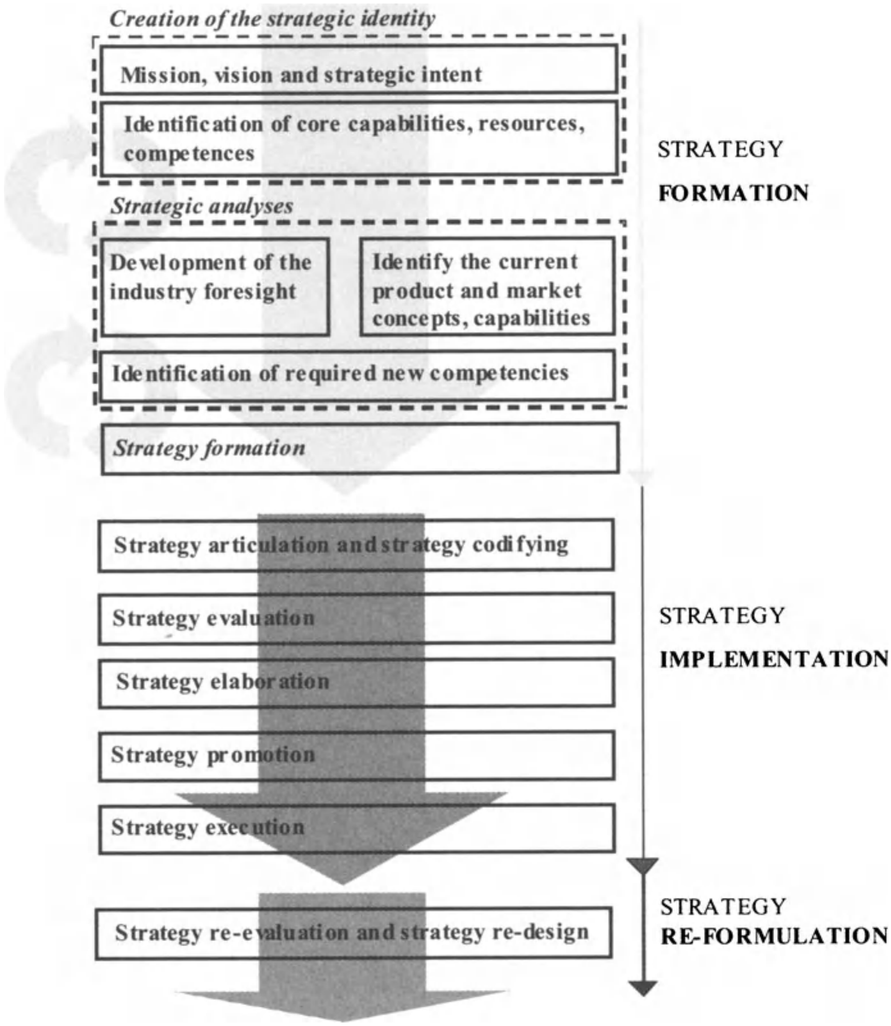


Fig. 4.5. The strategy framework

In spite of the comprehensive strategic management literature, managers can hardly find general guidelines or reference models, which could help manage the entire strategy life-cycle.

Also, a company needs to develop a view on its strategy, which recognises the need for *more than an instrumentalist annual planning*. What is needed,

is a strategic architecture that provides a *blueprint for building competencies needed to dominate future markets.*
The strategy framework presented in Figure 4.5 defines the strategy life-cycle composed of three main stages of life-cycle:

- the strategy formation stage (definition of the company's strategic identity, strategic analyses and strategy formation),
- the strategy implementation stage (which incorporates strategy articulation and elaboration, evaluation and execution), and
- the strategy re-evaluation and re-design stage.

The presented strategy life-cycle model incorporates three life-cycle stages. The relationship between these stages and life-cycle phases of enterprise reference architectures – GERAM (IFIP–IFAC Task Force , 1999), PERA (Williams , 1994), etc. – may be understood as follows:

- The first stage of strategy formation is mainly spent on the identification, conceptualisation and specification phase of strategy.
- The second stage includes the design and implementation, where the strategic concept is translated from its conceptual form to pragmatic goals and projects, which are then put in place in the company.
- The third stage includes (a) the operation of present strategic processes, and (b) in parallel with this, strategy is from time to time renewed. This renewal may be due to the change in internal or external circumstances, and therefore initiated by significant events, or is practiced because the validity of the present strategy's premises is wearing off (thus periodic reassessment of the current strategy is needed).

For a strategy's success the efficient performance of all life-cycle stages seems to be equally important. A brilliant strategy, if it is poorly implemented, and a poorly deliberated and formulated strategy excellently implemented bring about the same problematic result.

4.4.1 The Strategy Formation Process

Strategy formation stage incorporates three main life-cycle activities (phases) - thus at this stage the activities listed below may have to be performed in parallel, and/or iteratively:

- the creation of the company's strategic identity by the definition of the company's core statements (mission and vision statements and strategic intent) and identification of core capabilities and resources,
- analysis of the current position (of accumulated capabilities, current market and product concepts, conditions in external and internal environment) and new potential opportunities (development of the industry foresight),
- the strategy formation.

Normally the process of strategy formation is a *highly iterative process*, where activities form different life-cycle phases continuously exchange information (for example, the definition of a strategic identity cannot be completed without the consideration of information about the current position of the company, future development trends, etc., which in turn might be acquired through analysis).
The authors believe that the creation of a strategic identity and the execution of strategic analysis would usually prove to be the most difficult, demanding and time consuming parts of the strategy formation process.
The development of strategic identity asks for an in-depth conceptual discussion, supported by analytical tools. In this conceptual discussion, the strategic identity is created and key strategic decisions are made. These decisions shape and influence the other two activities.
The definition of a strategic identity is very important to assure a coherent and consistent stream of actions when significant events emerge requiring the company to develop a response.

4.4.1.1 The Creation of the Strategic Identity

Every firm must develop its own identity. Part of this identity can be defined in some basic statements about the company, such as vision, mission, policies and value statements. These statements may be deeply integrated into the company's behaviour and culture.
The company's mission and vision, as the company's core statements are quite often written in a very general way and therefore do not reflect the idiosyncratic nature of the company.

4.4.1.1.1 Defining a Mission

The mission statement defines *who we are* and *what is the fundamental purpose of our existence.* The mission statement must answer some important questions that define the current company profile. E.g.:

- What is the company's positions on the market?
- What kind of products and services does the company intend to provide to the customers?
- What kind of customers or markets are targeted?
- What is the market context in which the company sees itself (is it a sole provider, OEM manufacturer, niche market competitor, what are the relationship to customers, suppliers, competitors)?
- How does the company want to be perceived by its customers and by players in its external and internal environment?

Table 4.1 presents two sample mission statements. The first example is a very generic mission statement, often found in practice. The second mission statement is an example of a more defined and expressive statement, which

gives more information about the company's identity, relationship to the market (global market producer, which considers the requirements of world-wide technical standards), the company's customers (innovative and reliable products) and products (OEM product producer).

Table 4.1. Mission statement example

Example of a generic mission statement:
We are an electronic component manufacturer for word markets.
Improved mission statement:
We are an OEM electronic manufacturer with an international supplier base, manufacturing specialist components for global white goods, a company with innovative designs, concentrating on high reliability low whole-of-life cost products.

4.4.1.1.2 Having a Vision

A vision statement is an answer to the question "What do we want to be known for in 5 to 10 years". A vision statement should paint a picture of the company's *future environment* and *opportunities* (products/services) and characterises the company's *future competencies* and *capabilities.* It also should contain answers how the future is imagined and why it is necessary to be there.

The vision does not necessarily have to be a futuristic idea and it *must be considered feasible.* Any vision that is not based on a solid factual foundation is likely to be simply a fantasy.

Although the vision is not a plan, but a well-articulated idea, it should provide a focus for the company's future evolution and development.

To be able to formulate a strategy that shapes the future of an enterprise, management should find answers concerning the *ambiguous, complex* and *uncertain business reality* (characterised by the company's position today and its future state) and develop a *causal understanding* of its resources and capabilities (Hamel and Pralahad , 1994), e.g.:

- how would the industry be different ten years from now? How is this understanding shared by senior management?
- what is the basis for the company's competitive advantage today and in the future?
- is the company more a rule-maker than a rule-taker within the given industry?
- in what end-product markets does the company participate today and in which ones does it plan to do so in the future?

- what are the dangers posed by new, unconventional rivals?
- are potcntial threats to the current business model identified and widely understood?

4.4.1.1.3 Declaring Strategic Intent

Beside the long-term objectives defined by the company's vision, the company needs some more *tangible* and *short-term objectives* (coordinated and synchronised by the long-term ones), which provide emotional and intellectual energy for employees in their journey into the future.
'While the strategy defines the way to the future, an ambitious and compelling strategic intent represents this tangible goal, an animating dream that energises a company' (Hamel and Pralahad , 1994).
Strategic intent is a particular point of view about the log-term market or competitive position that a firm hopes to build over the coming decades (the sense of direction and destiny).
Therefore, in declaring a strategic intent, we try to identify clear corporate *challenges that focuses everyone's attention* on the next key advantage or capability building (e.g. entry to the US market, ultimate quality improvement, win over the main competitor, etc.)
With the explicit definition of the strategic intent, we ensure some 'cumulativeness' on a month-by-month and year-by-year decisions.
Strategic intent defines the company's priorities. Namely, many firms find themselves behind on cost, quality, cycle-times, customer service, attempt to push everything through in advance simultaneously and then wonder why progress is so painfully slow. No single business department can attend to all improvement goals and tasks at once.
As bad as having no clear goal, it is equally problematic having too many goals; therefore it is strategically important to give priority to goals and tasks of key importance.

4.4.1.1.4 Identifying Core Competencies and Core Products

A firm cannot actively manage its core competencies if those competencies are not recognized.
Irrespective of the strategic importance of core competencies, management could be confused about what is and what is not a core competence.
Typically, the first attempt to define core competencies produces an extended 'laundry list' of skills, technologies, and capabilities – with some proposed to be regarded as 'core' but most of them not really qualifying when investigated more deeply.
As widely recognized, a competence is a bundle of skills and technologies, rather then a single discrete skill or technology (e.g. the core competence of achieving fast cycle-times requires a broad range of underlying capabilities in the design that maximises commonality across a product line, flexible manufacturing and excellent supply management). To define a core competence,

it is necessary for it to *cluster* and *aggregate skills and technologies in some meaningful way* (meaningful in terms of being able to produce some important product or service).

The authors find that companies typically fall into one of several traps when attempting to identify core competencies. One of the most frequent traps is delegating the core competencies identfication task to the technical community. According to Hamel and Pralahad (1994), core competencies are the soul of the company and as such, their management must be an integral part of the management process of company executives.

As presented in Section 4.3.3, companies could posses many competencies, of which some are core and some are non-core. But how to make the distinction? The first criterion is whether the activities that are part of the competence really *contribute to long-term corporate prosperity.* The second criterion of being a core-competence is that the competence must 'pass' the tests and meet the criteria below:

- customer value – a core competence must make a disproportionate contribution to customer-perceived value,
- competitor differentiation – the capability must be competitively unique,
- extendibility – a core competence is not merely the ability to produce the current product configuration (however excellent that product line may be), but it also must be able to be used as a basis of potential new products.

Furthermore, the definition of core competencies requires the definition of *competence holders.* Visibility of a firm's resources that are the basis of a core competence is vital if they are to be fully exploited and easily redeployed.

Core products are the material manifestation of a company's core competencies (Mintzberg et al. , 1999). Therefore, the control over core products is essential to sustain the company's competitive position and in some cases its control over competitors. For this reason, many companies seek to sell core-products to other companies in order to create dependency and thus a position of control.

This virtual market share, revenues and experience accumulation allow the company to accelerate its core competence building effort, to open new markets or extend the volume of the existing market.

4.4.1.2 Strategic Analysis

A strategic analysis sub-phase incorporates different activities, and uses analytical tools and methodologies to define the current company's concept (capabilities, resources, products and markets), as well as to match it with future opportunities.

4.4.1.2.1 Industry Foresight

Why is industry foresight so important? Ultimately, the most important reason is the need to *support the long-term process of capability development.* To be prepared and take a share in the emergence of new business opportunities, a company must recognize the direction of its industry evolution and identify new requirements. Since leadership in industries is seldom built in anything less than 10 to 15 years, industry foresight is essential.
If a company would like to control its own destiny, it must understand how to control the evolution of its industry and proactively shape the future of it. Industry foresight looks at how the world could be different and the industry shaped in five or ten years. Hamel and Pralahad (1994) state that industry foresight and intellectual leadership requires a *deep understanding of technological, demographic, regulatory, life-style changes* that could be used to transform an industry's boundaries and shape tomorrow's opportunities.
Strategy is about competing both *within* today's-, and *for* tommorrow's industry structure.
How is it possible to develop industry foresight? Industry foresight is the product of many people's vision. The creative process of creating foresight may extract ideas from the answers to some conventional questions:

- whose product concept will ultimately win out?
- which are the potential customer benefits to be provided in the future?
- which standards will be adopted?
- how will coalitions form?
- how does one increase its ability to influence stakeholders in the industry / market?

However, unconventional questions should also be addressed, because they might unlock a new competitive space and enlarge the horizon of foresight.
CEOs aren't the only ones with industry foresight; their primary role is to capture and exploit the foresights that exist throughout the organisation; they are in fact responsible for ensuring the development of the *capability* to create industry foresight.
Industry foresight could be created when the future comes into view, e.g., when technologists are imbued with marketing imagination and marketers have developed deep insights into technology trends and demands through e.g. *empathy* with future human needs. For example, Honda matches the age of its design groups to the age of buyers targeted by a particular model.

4.4.1.2.2 Identification of Current Product, Market and Capability Concepts

The identification and analysis of current product and market concepts and accumulated capabilities requires different analytical tools and the aggregation, consolidation and interpretation of various internal / external information sources.
Companies can use different standard analytical tools, such as:

- SWOT[5] analysis (in practice SWOT analysis is often oriented towards corrective actions, rather than to identify opportunities and advantages)
- benchmarking with the competitors (not just with traditional competitors but also with potential new-comers that may be getting close to developing a particular core competence – whether in the same, or different way to the company doing the analysis),
- market investigations or
- evaluation of the negotiation position of existing customers and suppliers, entry barriers, threat of the appearance of new substitutes, and competition - including the intensity of such competition (Porter's five competitive forces).

4.4.1.2.3 Identification of Required New Competencies

Companies, based on the results of sometimes extensive analysis and research, can identify capabilities which have to be improved or developed within the scope of the current market and product concepts, or which are needed to dominate future markets and new emerging industries.
In this phase, a simple competence/market matrix (refer Fig. 4.6) proposed by Hamel and Pralahad (1994) may be used.

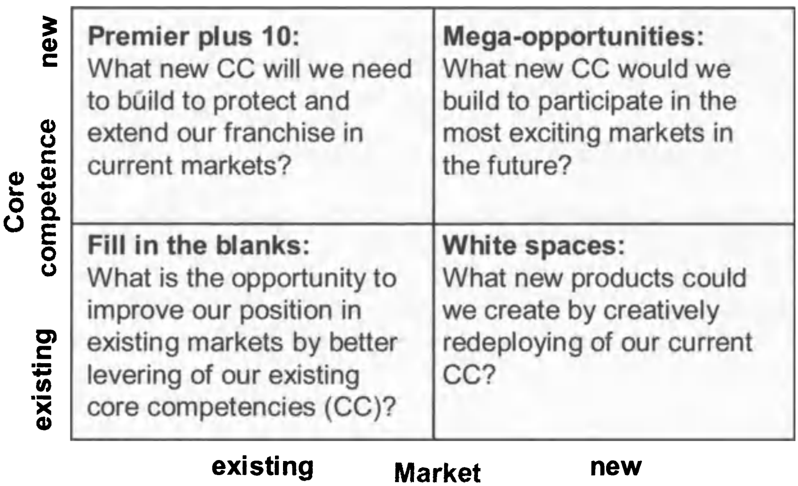


Fig. 4.6. Competence – market matrix for setting specific competence acquisition and deployment goals

Once management starts conceiving the company as a portfolio of competencies, a whole new range of potential opportunities may open up. The

[5] Strengths, Weaknesses, Opportunities and Threats

company should recognise that competition for core *competence leadership precedes competition for product leadership.*

The establishment of world leadership in a core competence area may take five or more years and requires consistency of all competence building efforts. Consistency depends first of all on a deep consensus on which competencies to build and support. Without such consensus, a company may fragment its competence building efforts as various business units would pursue their independent competence building agendas.

4.4.1.3 Strategy Formation

During the performance of the strategic analyses, many important strategic decisions would already have been made. Therefore, the strategy formation sub-phase provides a co-ordination, or synchronisation of decisions and integrates them into a cohesive strategy. In this phase, we have to

- create or confirm strategic directions,
- define the way to implement strategic directions (definition of migration paths), and
- prepare and investigate some possible scenarios.

To support the decision making process, some reference models can be used (more in the fashion of a checklist then a formal model at this stage), similarly as generic product and market strategies or organisational strategy types, that are presented in the remainder of this chapter.

4.4.1.3.1 Developing a Balanced Portfolio of Capabilities

Some companies have been capable to develop top-end discrete and specialised skills. However, to be able to launch successful final products what counts is not just the possession of discrete functional skills but the ability to integrate diverse functional skills from R&D to production (according to world-class levels of cost and quality), as well as marketing and sales (widespread distribution, marketing and service infrastructure).

In the absence (or shortage) of any of the key competence pillars, the company will be unable to fully exploit its investments made in areas of strength and leverage firm-specific resources.

Beside functional and cross-functional skills, the company must recognise the need for developing an integrative capability to improve or expand specific functional capabilities (such as production capabilities).

4.4.1.3.2 Having a Resource- and Capability Acquisition Agenda

Many of the most exciting new opportunities require the integration of complex capabilities (identified in the previous sub-phase). Therefore, the company has to develop an agenda to acquire resources and capabilities (possible

scenarios - own development, acquisition of a company which posses those capabilities, etc.).
Most management text books on innovation and product development assume that the company controls most of the resources needed for the commercialisation of that innovation; but in fact such as assumption is increasingly likely to be wrong.
To supplement internal resources with the resources outside of the formal boundaries of the firm requires the development of an integrative capability. In practice, we can find different forms of capability acquisition:

- tight links with suppliers to better exploit their innovation,
- sharing development risks with critical customers,
- borrowing resources from more attractive sources,
- participating in international research consortia,
- cooperation with competitors – usually in the early stages of market evolution

4.4.1.3.3 Strategically Positioning the Company

Firms can be categorised into some archetypes, based on the relationship between strategy types, distinctive marketing competencies and organizational performance. One of the well-known archetype models has been developed by Miles and Snow (1978), based on a theory of 'Strategy, Structure and Process'.
According to entrepreneurial orientation (organizational readiness for risk-taking and for the development of new products), technological orientation and administrative orientation (types of control employed in the organization), a company can be classified into one of the following four organizational configurations (according to Miles and Snow (1978), refer Fig. 4.7):

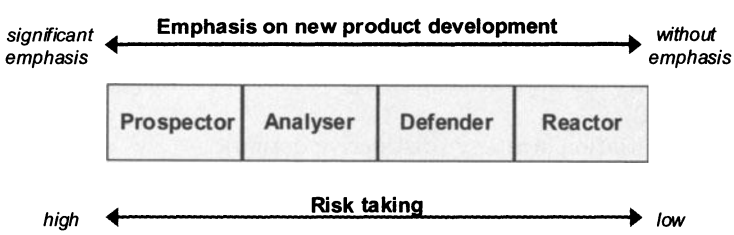


Fig. 4.7. Miles and Snow 's (1978) typology of organisations

- Prospector - always seeks new opportunities and experiments with innovations. Because of the uncertainty in the development of new product concepts or industries, this type of company takes high risks. Prospectors,

emphasizing technological leadership, are heavily investing in technology. Prospector organisations promote creativity and flexibility; they are often decentralised, capable of fast reaction, and able to redefine existing products and markets.

- Analyser - imitates, or copies the success of prospectors, with the ability to respond to prospector initiatives and to maintain operating efficiency at current markets (defender behaviour). Analysers follow an intermediate strategy; they are more careful than the prospectors and decide upon investments in new technologies only after a thorough analysis of the possibilities.
- Defender - seeks stability with a limited number of products, in a well-defined industry. Defenders are more conservative in their investment in technology and focus on technological areas directly related to their line of business.
- Reactor - responds to environmental pressures and events without a clear focus or rationale. Reactor organisations have relatively little strategic vision and their decisions are usually more short-term oriented than long-term.

The authors would like to caution the reader to the fact that none of the presented strategic organisation types is a guarantee of-, or sole impediment to business success. In practice, evidence of successful companies may often be found in each of the prospector, defender and analyser types[6]. All three of the above-mentioned types usually rely on a well-thought-, formulated- and defined strategy - in contrast with the reactor type of organisation.

4.4.1.3.4 Having Generic Product Strategies

When major strategic directions are defined and confirmed, one can proceed with the definition of strategy migration paths and scenarios, and employ at least some elementary product and market strategies.
In the comprehensive marketing literature, some elementary types of product strategies can be found, such as (Mintzberg et al. , 1999):

- low cost or price differentiation strategy (e.g. low volume, commodity type of production),
- image differentiation strategy (distinctive design),
- support differentiation strategy (provision of a quality after sales service),
- quality differentiation strategy (more reliable and durable products),
- design differentiation strategy (added, improved product functionality).

or strategies that elaborate or extend the range of products offered, e.g.:

- penetration strategy (same product sold more intensively on the same market – through increased advertising)

[6] **NB**it is highly unlikely that a company that finds itself consistently in a reactor position could be successful in the long run

- bundling strategy (selling different products together – such as HW and SW together)
- market development strategy (targeting same product types on the market)
- product development strategy (targeting new products on same markets)
- diversification strategy (targeting different products at different markets), etc.

For a company to be capable of defending or improving its current market position and to achieve long-term success, it should manage and balance its product portfolio. Figure 4.8 shows a matrix defining four categories of products according to market share and expected growth.
A balanced product portfolio provides constant cash flow and consequently the firm's ability to invest into the development of new products, which (hopefully) will be major market players in the future (and future 'cash cows').
Such a balanced portfolio incorporates (according to Handerson (1979), refer Fig. 4.8):

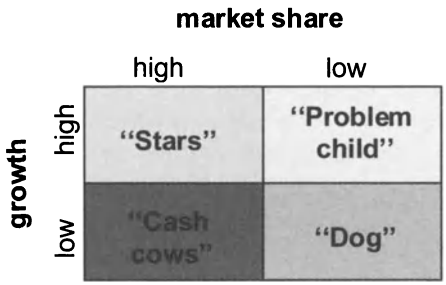


Fig. 4.8. The balanced product portfolio – each area in the matrix represents a product type (the 'Boston Box, Boston Consulting Group):

- stars, whose high share and growth guarantee the future (recommended strategy: growth; add resources and build the business further based upon market projections),
- cash cows, which supply funds for future growth (recommended strategy: stability or modest growth; maintain benefits of strong cash flow while keeping resource investment to the minimum),
- problem child, waiting to be converted into stars (recommended strategy: growth or retrenchment; apply resources to accomplish positive turnaround or pull back if the outlook is poor).
- dogs, which are not necessary. Dogs are cash traps (recommended strategy: retrenchment; divest, sell, liquidate the corresponding type of business activity to eliminate resource drain).

4.4.1.3.5 Developing Generic Market Strategies

As an extension of product strategy it is necessary to have a *market* strategy. Markets are differentiated by:

- *size and divisibility* (mass market – large and homogenous, fragmented market – many small niches, segmented market – different demand segments, thin market – few occasional buyers)
- *location* (local, regional, global) and
- *stage of evolution*:
  - emerging markets – young, not yet clearly defined, with unstable competition;
  - established markets – clearly defined, with unstable competition;
  - eroding markets, with stable competition;
  - erupting markets – undergoing destabilising changes, with multi-point competition).

When the company creates a new product and launches it on the market, there are two different scenarios according to the fit between product and market:

- natural fit, where the product creates the market or the market encourages the development of the product, or
- forced fit (usually real world situation), which requires reinforcing mechanisms (to improve the fit), or isolating mechanisms (to protect the fit).

A company can play (according to the market situation and market characteristics) one, or several generic market strategies. Some possible strategies are:

- fortifying strategy (build up barriers or shelters around the product – patent protection, long-term contracts with customers, etc.);
- collaborative strategy (cartels);
- burrowing strategy;
- packing strategy;
- attack strategy (frontal attack, lateral attack – undermining – taking away loyal customers),
- etc.

4.4.2 The Strategy Implementation Process

After the completion of the strategy formation phase, the strategy could be still high-level defined in form of major strategic directions and strategic scenarios.
For the strategy to be successful, it must be translated into:

- the strategic plan, which defines major actions to be taken and the strategic resources needed to carry out actions,

- performance indicators to measure progress toward the defined strategic objectives.

The selected strategy also has to be promoted and recognised at all levels of the company hierarchy, so as to allow for any individual in the company to make and recognise his/her own space and contribution in the journey to the future.

4.4.2.1 Strategy Articulation and Codification

Strategy formation, as a creative and innovative processes, can deliver a strategy, but it may still be only something that is internalised in the heads of company executives and may not be well-formalised either.
Therefore, the task of strategy articulation is the externalisation of the essence of the strategy and its formulation in an explicit way. In the phase of strategy *codification*, the strategy has to be translated into:

- strategic objectives that are tangible and measurable (to provide a base for monitoring the success or failure of the strategy's implementation),
- actions and projects through which strategies are achieved
- key-success factors and performance indicators to monitor the strategy's performance and its efficiency (employment of financial and non-financial as well as quantitative and qualitative indicators),
- organisation policies, which are rules and guidelines that express the limits imposed on actions that should occur. These rules often take the form of contingent decisions for resolving conflicts among specific objectives.

In setting performance indicators, many executives are too focused and interested in the development of a set of financial indicators, which show the growth of the revenue, profit rate, market share, etc.
Traditional *financial indicators* reflect the result of the company's previous activities and efforts (they can also be called *lagging* indicators), therefore supplemental *non-financial indicators* have to be developed and employed.

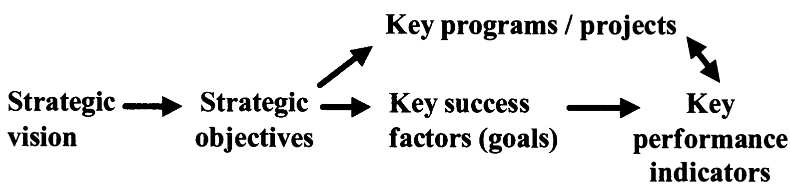


Fig. 4.9. Balanced Scorecard (BSC) chain

Non-financial indicators characterise better the structural development of the company, the organisation's potential and its corporate health. Therefore, those indicators could be considered as *leading* indicators.

Kaplan and Norton (1996), in their Balanced Scorecard (BSC) methodology, identify four categories of performance indicators (learning and growth, business processes, customer relationship and financial performance – as shown in Fig. 4.9), while the European Foundation for Quality Management (EFQM , 1999) organizes 32 sub-criteria into 9 major criterion groups (refer Fig. 4.10). BSC also allows the creation of a tangible link between the organisation's vision and its translation into strategic objectives, key success factors, key projects and performance indicators.

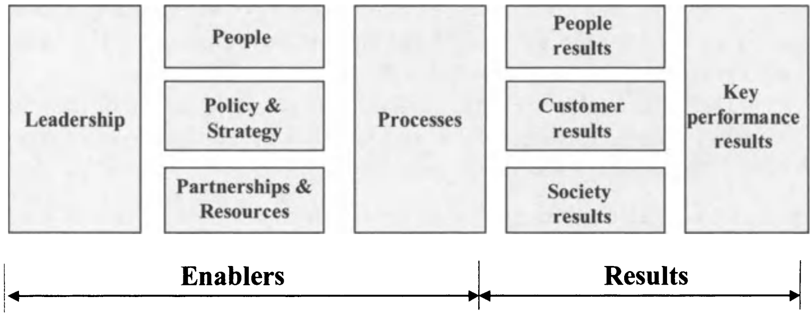


Fig. 4.10. EFQM model (each area represents a number of performance indicators)

4.4.2.2 Strategy Evaluation

Once the strategy is articulated and codified, it could be evaluated against the following simple criteria:

- Consistency between objectives, goals, actions, policies and organisational values; consistency is necessary so as to provide coherence to the organisation's actions and to decrease the amount of interventions for coordination;
- Consonance of the strategy; the strategy should represent an adaptive response to the external environment and to the critical changes occurring within it;
- Advantage: the strategy must create a competitive advantage in the selected area of activities; competitive advantages are the advantages that are most telling, enduring and difficult to duplicate. Competitive advantage can derive from skills (which are acquired through learning by doing), resources (which are specialised to the firm and slowly build up over the time) or position (for instance first movers or reinforcers);
- Feasibility of the strategy.

4.4.2.3 Strategy Elaboration

Strategy elaboration is dedicated to *transform the strategy into executable* and *operational plans*, presented in form of a *strategic plan* and subsequent *annual plans.*

Plans are a *translation of strategic objectives into tactical and operational goals*, and incorporate a detailed definition of actions and projects, task allocation, required resources, budgeting, etc.

The company's overall strategic plan must be further decomposed into *detailed plans for business units*, departments, etc. all down to the explicit definition of individual goals of employees (employment of the concept of a management by objectives).

A frequent mistake at this stage is when company management attempts to complete this decomposition alone, while in fact it is necessary that each group, or individual should be able to do this decomposition for themselves, so as to operationalise the strategy.

Strategy elaboration should also develop a manager's '*control panel*' - to monitor and control the strategy performance. This 'control panel' defines the objectives and goals to follow, the performance indicators to monitor, and the sources of information from which these indicators can be derived. The development of the 'control panel' in fact is tantamount to the development of a matching strategic management information system, which provides feedback about progress in the execution and contribution of projects/programmes and about the state of achievement of strategic objectives and goals.

4.4.2.4 Strategy Promotion

Strategy is of little value if it is not widely debated and ultimately understood by all employees. Therefore, organisations usually launch an extensive and deliberate 'advertising' campaign for the recognition of the new strategy or significant changes in the existing strategy.

Strategy should be understood at all hierarchical levels and should give space for contribution to every employee in the company. The strategy promotion is important in the phase of strategy recognition and understanding, however the *provision of feedback* about the strategy's success is also of great significance. Regular reports should give information about the results of the strategy implementation and about the achievement of strategic objectives.

Management also should encourage the recognition and rewarding of individual contributions in the achievement of strategic objectives.

4.4.2.5 Strategy Execution

Strategy execution incorporates actions for the strategy implementation and continuous control of strategy performance. In this sub-phase organisations

can employ project management methodologies for the definition and set-up of strategic projects as well for their execution and control.
Subsequent chapters of this book discuss activities that can ultimately lead to the implementation of strategic change.

4.4.3 Strategy Re-evaluation and Re-formulation

Due to constant changes in the internal and external environment of the company and certain assumptions and forecasts used in the strategy formation process, strategy must be regularly reviewed and validated. Strategy re-evaluation is necessary at regular intervals, as well as a response to a significant event. Thus, the evaluation may be triggered:

- by a regular annual re-evaluation process,
- as a consequence of unsatisfying business results,
- as a consequence of serious mismatch between the expected and actual performance indicators, or
- as the result of changes in the company's top management.

A good strategy does not need constant re-formulation. What is needed is to assure that there is adequate:

- constant exploitation of productive opportunities;
- incremental development of capabilities;
- reaction to emerging events, and
- continuous flow of actions in order to foster organisational learning.

4.5 Conclusion

Successful companies are in the heart of contemporary society's well-being. Understanding of business success or failure is crucial for creating a successful corporate strategy and is the most important task for managers.
For the creation of an efficient strategy, the employment of prescriptive tools, methodologies and models is insufficient and must be accompanied with in-depth conceptual knowledge. Managers should be aware that each business strategy is unique and that a strategy is neither 'wrong' nor 'right' in any absolute sense. Strategy is about building the competencies needed to dominate future markets.
Strategy can *form* (emerge in response to an evolving situation) as well as be *formulated* (it can be brought about deliberately, through the process of formation, followed by implementation).
It is the managers' main responsibility to be the main protagonists for the creation of company future, and to steer the company to get to it. Therefore, executives are responsible for the management of the strategy through the entire strategy life-cycle.

If a company wants to create a good strategy, it must *support innovative thinking* and *reward unorthodoxy.*
It is to be noted that strategic planning – the notion often used in the business environment synonymously with strategy formation – is not about creating a strategy, but about programming a strategy already created (Mintzberg , 1993). Planning is essentially analytical in nature, based on decomposition, while strategy creation is a process of synthesis, creative and innovative in its essence.
This Chapter has offered a strategy framework which incorporates the conceptual knowledge of the Resource Based View (RBV), intertwined with some well-known tools in the domain of strategy making.
Future attempts should be focused on the applied dimension of conceptual knowledge by providing frameworks or reference models where this knowledge is structured and help is more formally described in this often ambiguous organizational phenomenon.

References

Amit, R., Schoemaker, P. J. H. (1993) Strategic assets and organizational rent. Strategic Management Journal. 14(1) pp. 33-46

Barney, J. B. (1986) Strategic factor market: expectations, luck, and business strategy. Management Science. 32(10) pp. 1231-41

Barney, J. B. (1991) Firm resources and sustained competitive advantage. Journal of management. 17(1) pp. 99-120

Black, J.A. and Boal, K.B.(1994) Strategic resources: Traits, configurations, and paths to sustainable competitive advantage. Strategic Management Journal [SMJ], 15(??), 131 - 148.

Brandenburger,A.M., Nalebuff,B.J (1995) The right game: use game theory to shape strategy. Harvard Business Review. pp57-71

Brown, S. L., Eisenhardt, K. M. (1998) Competition on the Edge: Strategy as Structured Chaos. Boston (MA) : Harvard Business School Press

Coase, R. E. (1937) The nature of the firm. Economica. Vol.4, pp386-405

Collis, D. J. (1994) Research note: how valuable are organizational capabilities? Strategic Management Journal. Vol. 15, pp143-52

Conner, K. R. (1991) A historical comparison of resource-based theory and five schools of thought within industrial organization economics: do we have a new theory of the firm? Journal of Management. 17(1) pp121-154

Cyert, R. M., March, J. G. (1963) A Behavioural Theory of The Firm. New York : Prentice Hall

Davenport, H. T. (1993) Process Innovation. Boston (MA) : Harvard Business School Press

Didrickx, I. and Cool, K. (1989) Asset stock accumulation and sustainability of competitive advantage. Management Science. 17(1) pp121-154

Dosi, G., Nelson, R. R., Winter, S. G. (2000) The Nature and Dynamics of Organizational Capabilities. Oxford : Oxford University Press

Edelman, G. M. (1987) Neural Darwinism: The Theory of Neuronal Group Selection. New York : Basic Books

EFQM (1999) The EFQM Excellence Model. Brussels : European Foundation for Quality Management

Eisenhardt, K. M., Martin, J. A. (2000) Dynamic capabilities: what are they? Strategic Management Journal. Vol. 21 (Special Issue) pp1105-1121

Fujimoto, T. (2000) Evolution of manufacturing systems and ex post dynamic capabilities. in Dosi, G., Nelson, R. R. and Winter, S. G. (Eds) The Nature and Dynamics of Organizational Capabilities. Oxford : Oxford University Press pp244-280

Ghemawat, P. (1999) Strategy and the Business Landscape: Text and Cases. Harlow : Addison-Wesley

Gulati, R. (1998) Alliances and network. Strategic Management Journal. 19(4) pp293-317

Hall, R. (1992) The strategic analysis of intangible resources. Strategic Management Journal. Vol. 13, pp135-144

Hamel, G., Prahalad, C.K. (1994) Competing for the future. Boston (MA) : Harvard Business School Press

Handerson, B.D. (1979) Henderson on Corporate Strategy. Cambridge (MA) : Abt Books. pp163-166

Hannan, M.T., Freeman, J. (1984) Organizational Ecology. Cambridge (MA) : Harvard University Press

IFIP–IFAC Task Force (1999) GERAM: Generalised Enterprise Reference Architecture and Methodology. Version 1.6.3. http://www.cit.gu.edu.au/~bernus

ISO (2002) SIST ISO 9001:2000 Quality management system requirements. http://www.iso.ch

Kaplan, R. and Norton, D. P. (1996) The balanced scorecard: translating strategy into action. Boston : Harvard Business School Press

Lane, D., Malerba, F., Maxfield, R., Orsenigo, L. (1996) Choice and action. Journal of Evolutionary Economics. Vol 6, pp43-76

Mahoney, J. T., Pandian, J. R. (1992) The resource-based view within the conversation of strategic management. Strategic Management Journal. 13(5) pp363-80

Makadok, R, (2001) Toward a synthesis of the resource-based and dynamic-capability views of rent creation. Strategic Management Journal. 23(5) pp387-401

Markides, C.C. (1999) A dynamic view of strategy. Sloan Management Review. 40(3) pp55-63

Miles, R.E., Snow, C.C. (1978) Organizational strategy, structure and process. New York : McGraw-Hill

Miner, A. S., Bassoff, P., Moorman, C. (2001) Organisational improvisation and learning: a field study. Administrative Science Quarterly. Vol.46 pp304-37

Mintzberg, H. (1978) Patterns in strategy formation. Management Science. 24(9) pp934-948

Mintzberg, H., Waters, J.A. (1985) Of strategies, deliberate and emergent. Strategic Management Journal. 6(3) pp257-272

Mintzberg, H (1993) The Rise and Fall of Strategic Planning. New York: The Free Press.

Mintzberg, H. (1998) Strategy Safari: A Guided Tour Through Wilds of Strategic Management. London : Prentice Hall

Mintzberg, H, Quinn, J.B., Ghostal S. (1999) The strategy process – revised European edition. London : Prentice Hall

Nelson, R., Winter, S. (1982)An Evolutionary Theory of Economic Change. Cambridge (MA) : Belknap Press

Penrose, E. T. (1959) The Theory of the Growth of the Firm. New York : Wiley

Porter, M.E. (1980) Competitive Strategy. New York : Free Press

Porter, M. E. (1996) What is strategy? Harvard Business Review. 74(6) pp61-78

Prahalad, C.K, Hamel, G. (1990) The core competence of the corporation. Harvard Business Review 68(3) pp79-91

Shapiro, C. (1989) The theory of business strategy. Rand Journal of Economics. Vol.20, pp125-137

Schendel, D. (1994) Introduction to the summer 1994 special issue strategy: search for new paradigms. Strategic Management Journal. Vol. 15 (Special Issue) pp1-4

Stalk, G. Evans, P., Schulman, L. E. (1992) Competing on capabilities: the new rules of corporate strategy. Harvard Business Review. 70(2) pp57-69

Teece, D. J., Pisano, G., Shuen, A. (1997) Dynamic capabilities and strategic management, Strategic Management Journal. 18(7) pp509-33

Verona, G. (1999) A resource-based view of product development. Academy of Management Review. 24(1) pp132-42

Wernerfelt, B. (1984) A resource-based view of the firm. Strategic Management Journal. 5(2) p171-80

Whittington, R. (1993) What Is Strategy – And Does It Matter? London : Routledge

Williams, T.J. (1994) The Purdue Enterprise Reference Architecture. Computers in Industry. 24(2-3) pp141-158

Williamson, O. E. (1985) The Economic Institutions of Capitalism. New York : Free Press

Winter, S. G. (2000) The satisfying principle in capability learning. Strategic Management Journal. Vol. 21 (Special Issue) pp981-96

5

LEADERSHIP: BETTER RELATIONSHIPS THROUGH BETTER COMMUNICATION

Hugh Mackay

Mackay Research - NSW (Australia)

5.1 Introduction [1]

This Chapter is significantly different in style from the others, and with good reason. First, this text is reprint of a chapter written by the author[2] as part of a book on communication. Second, it is about human relationships that so much influence the success or failure of management's effort to carry out change.

Even if individuals in the organisation see that the arguments for change stand to reason, there is a natural tendency to defend the *status quo* – either through voicing opposition, or by actions. It is natural to defend the present arrangements and to keep doing things the same way they have been done in the past. Change implies learning, negotiating, abandoning routines, establishing new authorities and responsibilities and losing some old ones. All of these take people out of the *comfort zone*. As opposed to this, a 'no change' stance takes the least effort and minimises short term risk – 'if it worked yesterday it will work tomorrow' – so, even if things are not working perfectly in the present, people know how to deal with problems on the day-to-day level. Perhaps people can be convinced by pure logical arguments that change is necessary, but the tendency to avoid it remains very strong. *How can top management or any leader of a change effort negotiate such an obstacle?*

The methodology to address the relationship of humans to change calls for approaches that are different from those that are appropriate for technical problems. Communication is at the heart of successful management, and the practice of Enterprise Architecture must be very good at it in order to successfully change enterprise processes, technologies, the organisation and involved human relationships. While management has the power to carry through many projects, success can be achieved only if people in the organisation embrace the solution.

[1] by the Editors

[2] original appeared in (Makay,1998). Reprinted with permission

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

Some individuals in the organisation will be actively participating in the change process, and their attitudes will be different - from championing, accepting, co-operatively supporting, to actively opposing it. Most of the of the others will take a *passive* role – observing, tolerating, accepting, going along with, or passively obstructing. Management would like to see members of the organisation ultimately change their mindset, and this change is the hardest of all to achieve, notwithstanding the technical problems that the enterprise faces in the process. This C hapter investigates how people change their minds and their behaviour, and understanding this relationship contributes to the successful management of change.

5.2 Changing People's Minds

'You knew it must come to this, sooner or later, Toad,' the Badger explained severely. 'You've disregarded all the warnings we've given you, you've gone on squandering the money your father left you, and you're getting us animals a bad name in the district by your furious driving and your smashes and your rows with the police. Independence is all very well, but we animals never allow our friends to make fools of themselves beyond a certain limit; and that limit you've reached. Now, you're a good fellow in many respects, and I don't want to be too hard on you. I'll make one more effort to bring you to reason. You will come with me into the smoking room, and there you will hear some facts about yourself; and we'll see whether you come out of that room the same Toad that you went in.' He took Toad firmly by the arm, led him into the smoking room, and closed the door behind them. 'That's no good!' said the Rat contemptuously. 'Talking to Toad'll never cure him. He'll say anything.' They made themselves comfortable in armchairs and waited patiently. Through the closed door they could just hear the long continuous drone of the Badger's voice, rising and falling in waves of oratory; and presently they noticed that the sermon began to be punctuated at intervals by long-drawn sobs, evidently proceeding from the bosom of Toad, who was a soft-hearted and affectionate fellow, very easily converted – for the time being – to any point of view. After some three-quarters of an hour the door opened, and the Badger reappeared, solemnly leading by the paw a very limp and dejected Toad. His skin hung baggily about him, his legs wobbled, and his cheeks were furrowed by the tears so plentifully called forth by the Badger's moving discourse. "Sit down there, Toad,' said the Badger kindly, pointing to a chair, 'My friends', he went on, 'I am pleased to inform you that Toad has at last seen the error of his ways. He is truly sorry for his misguided conduct in the past, and he has undertaken to give up motorcars entirely and forever. I have his solemn promise to that effect.' 'That is very good news,' said the Mole gravely. 'Very good news indeed', observed the Rat dubiously, 'if only – if only –' He was looking very hard at Toad as he said this, and could not help thinking he perceived something vaguely resembling a twinkle in that animal's still sorrowful eye. 'There's only

one thing more to be done', continued the gratified Badger. 'Toad, I want you solemnly to repeat, before your friends here, what you fully admitted to me in the smoking room just now. First, you are sorry for what you've done, and you see the folly of it all. There was a long, long pause. Toad looked desperately this way and that, while the other animals waited in grave silence. At last he spoke. 'No!' he said a little sullenly, but stoutly; 'I'm not sorry. And it wasn't folly at all! It was simply glorious!' 'What?' cried the Badger, greatly scandalised. You back-sliding animal, didn't you tell me just now, in there–' 'O, yes, yes, in there', said Toad impatiently. 'I'd have said anything in there. You're so eloquent, dear Badger, and so moving, and so convincing, and put all your points so frightfully well – you can do what you like with me in there, and you know it. But I've been searching my mind since, and going over things in it, and I find that I'm not a bit sorry or repentant really, so it's no earthly good saying I am; now, is it?' 'Then you don't promise', said the Badger, 'never to touch a motorcar again?' 'Certainly not!' replied Toad emphatically. 'On the contrary, I faithfully promise that the very first motorcar I see, poop-poop! Off I go in it! 'Told you so, didn't I?' observed the Rat to the Mole. 'Very well, then', said the Badger firmly, rising to his feet. 'Since you won't yield to persuasion, we'll try what force can do.'

The Wind in the Willows
Kenneth Grahame

Almost all communication involves some degree of persuasion. Someone asks you the time: they are persuading you to tell them. If you do, you are persuading them to accept what you say is true. When you seem to be trying very hard not to change someone, you may still be being persuasive: 'Stay as sweet as you are' is an appeal to resist change.

What about a simple 'goodnight'? Neutral? Not quite: it doesn't have to be laden with passionate connotations to carry some persuasive implications: recognise me as a polite person; remember me kindly, perhaps. At the very least, it would express the intention to create a particular sort of impression on the listener (which would be different from saying, for example, 'Get out of my sight, you old goat').

I don't want to push this too far. I simply want to register the fact that the distinction, which seems to exist between 'simple' communication and persuasion, may not be so easy to define. It is tempting to say that persuasion is about trying to influence another person to think, feel or act in a certain way, whereas communication is simply about the sharing of meaning with someone else – even when no influence is intended – but that's a rather blurred distinction.

What about the filing of a routine report? Even with a simple record of information – a set of statistics – the author wants to reassure you that the report is credible and its author competent and authoritative, and the way the report is presented will no doubt incorporate some signals designed to persuade you of that.

Even a desultory conversation around the dinner table has some persuasive content. 'Have a little more beef.' 'Wasn't it a lovely day?' 'Have you done your homework?' 'I think I'll do a bit of reading' (implication: 'so leave me in peace, please').

Messages which seem to be a simple case of keeping a relationship 'ticking over' are not always quite as simple as they look. When we want to 'keep in touch', there is still a hint of persuasion in wanting to keep other people interested in us, supportive of us, or prepared to keep their side of the relationship alive.

We often think of advertising as being a persuasive form of communication – and so it is. But the astute advertisers know that their most effective strategy is not to concentrate on changing people's minds, but to persuade through reinforcement: to strengthen the bonds of loyalty among existing consumers and to encourage others to try brand X because of how they feel now (rather than trying to get them to feel differently first). So, is 'preaching to the converted' a form of persuasion? Is further reinforcement of an existing attitude a form of 'changing people's minds'? (It is certainly a form of influence ... and that's what we normally mean by persuasion.)

One difference between communication and more obvious forms of persuasion may be that, in some circumstances, we really *want* to influence another person, whereas in other cases (such as 'Have a little more beef') we may not care one way or the other. Whatever the difference might be in marginal cases, it is obvious that there are many times when we very definitely want to *change* the attitudes or behaviour of another person: we want to modify – rather than merely reinforce – the cage. We may call it training. We may call it correction. We may call it advice. Whatever we call it, persuasion is clearly involved. One person is trying to 'change the mind' of another person.

Not all attempts to influence other people are benign: unscrupulous politicians, religious fanatics, con-artists, aggressive salespeople – anyone driven by a desire for personal power or commercial advantage – may well want to change other people's minds to serve their own ends rather than to benefit those whom they are trying to influence.

But it is often necessary to exert persuasive influence over other people *in their own interests*. Traffic authorities want to modify the behaviour of motorists in order to ensure their safety. Teachers want to influence their pupils in the interests of their own education. Parents want to socialise their children so they can take their places in adult society. Counsellors and health professionals want to help their clients to modify aspects of their attitudes or behaviour in order to overcome problems which may be interfering with the satisfactory conduct of their lives. Conductors want to influence the behaviour of their musicians in the interests of better music making. Sporting coaches want to influence their players to improve their performance.

The desire to influence other people is most obvious in the workplace, where people must be trained to do things in a certain way; guided to become more efficient or productive; shown how to adapt to new processes. For most of us

the world of work is a world of change and that change is ultimately brought about by people influencing each other to do things differently.

5.3 Defending the Cage

One of the myths about the process of personal influence is that, if you want to get someone to act differently, you must first get them to think differently. (Anyone who understands the cage knows why that is a myth).
We are most unlikely to change people's minds by asking them to change, simply because almost any direct attack on the cage is bound to be defended. Even the most sophisticated techniques of propaganda don't change people's minds in the direct way that is popularly imagined.
When the cage is attacked by people who are trying to change our minds, we have three favourite defences.

5.3.1 'Yes, but...'

The first is the *'Yes, but...'* defence which concedes the point of the attack, but smothers it in counter-arguments.
For example, the committed smoker hears all the anti-smoking arguments and says, 'Yes, but my father smoked sixty cigarettes a day and lived to the ripe old age of ninety. Anyway, I'm here for a good time, not a long time.'
The child who is being told that a particular TV program may not be viewed says, 'I know you don't like me watching that program, but all the other kids in my class watch it and, anyhow, I have done all my homework and there's nothing else to do except watch TV.'
The employee who is being told that her punctuality leaves something to be desired says, 'Yes, I am often late, but I get all my work done and I always take a short lunch break to make up for it. In any case, I think I achieve much more than some of the clock-watchers around here'.
The motorist, being warned about the hazards of drink-driving, says, 'Yes, I know what all those experiments about reaction-time are supposed to prove, but I personally find that my reflexes are actually sharpened by a couple of drinks. I feel more confident and that makes me a better driver'.
The patient who is being warned by his doctor about the dangers of a high-fat diet may say, 'I'm sure I could reduce the risk of heart disease if I cut down on fat but, on the other hand, I could be knocked over by a bus tomorrow. When your number comes up, your number comes up. In any case, it's all in your genes'.
The 'Yes, but....' defence is so spectacularly successful in resisting attempts to change our minds because it saves us from being drawn into the argument. It shifts the focus away from the troublesome proposition and focuses, instead on the unrelated proposition, which seems to have equal or greater weight. By retreating into the 'Yes, but...' defence, we never have to respond to

the attack: we mount a counter-attack of our own. The original argument is treated as if it is no longer relevant: we've moved on.

5.3.2 'What would they know?'

The second classic defence is the *'What would they know?'* defence. Once again, the listener chooses not to reply to the argument but tries, instead, to discredit the source of the argument.

This is a common strategy in politics where attacks on the character of a politician are thought to weaken the force of his or her arguments. Similarly, people who are in disagreement with research findings will often try to discredit the reputation of the researcher (and, by implication, the validity of the research) so that they don't have to respond to the findings.

Young people typically use the 'What would they know?' defence to protect them from the advice (or the criticism) of older people: 'What would they know? Things were so different in their day. The world has changed.'

When an academic offers some unpalatable advice to the community, the response may well be, 'What would he know? He lives in an ivory tower.'

When a child attempts to modify the attitudes of behaviour of a parent (for example, in relation to health issues, or adoption of new technology, or even in the interpretation of current affairs), the parent may well deflect the message with an attack on the immaturity of the child: 'What would you know? When you are older, you will understand these things.'

The young, enthusiastic engineer who is trying to convince factory workers of the value of new work practices may well run up against this particular defence: 'These blokes are all the same. Straight out of University. I've been doing this job for thirty years...I think I have got a pretty fair idea of what works and what doesn't. What would he know?'

5.3.3 'It couldn't happen to me'

The third defence against messages designed to change our minds is that *'It couldn't happen to me'*.

This is a favourite response when someone is threatening us with dire consequences if we go ahead with something we want to do; when people warn us of the dangers of a desirable course of action; when we hear of difficulties faced by other people who have attempted what we are planning to do.

A person is fired with enthusiasm for the idea of opening a gift shop in a new suburban shopping centre. Various friends and advisers are urging caution, quoting countless examples of small business enterprises which failed because of lack of adequate capital, lack of business training and expertise and lack of proper market research. But no: the person is so excited by the prospect of owning a shop that all arguments seem to fall on deaf ears.' I've heard all those stories you're telling me; I'm sure lots of people have made a mess of this kind of thing. But I just have a really good feeling about this – as soon as

I saw the shop, I knew I could make a go of it. All those things won't happen to me. I'm keen and I'm prepared to work hard. You'll see'.

A teenage girl is being warned by her friends about a heartless lothario who is showing some interest in her. 'He has a real reputation', they tell her. 'He'll be all over you like a rash, then as soon as he thinks he's won you. He'll be off after someone else. Don't be fooled by him. You'll be sorry'. Undeterred, the girl continues to encourage the boy's overtures. 'Don't worry,' she tells her friends, 'this is different. I'm sure he wouldn't treat me like those other girls. He's really nice. It won't happen to me'.

This defence allows us to protect our cages by denying the relevance of other people's experience. It also protects us when we are being threatened by horrifying messages – such as road safety or health propaganda – which are designed to change our minds through fear.

'People can lose a limb from smoking? Come off it. That couldn't happen to me. I jog every morning.'

The normal response to messages based on fear of horrific or even fatal consequences is either to 'switch off' or to reject the message as being too extreme to be relevant: 'Isn't that awful... it's too awful to happen to me... awful things like that don't happen to me... it couldn't happen to me.'

All three defences work as well as they do simply because our desire for reinforcement of the cage is so strong. Even when we can't deny the truth of an attack, we can divert it by relying on some other 'truth' entirely. When all else fails, we can argue that since the threatened event hasn't yet occurred, it isn't going to occur: 'After all, our own experience is the thing we should pay most attention to, isn't it?'

Resistance to attacks on the cage is part of a perfectly natural and predictable pattern of human behaviour. Knowing that, it's remarkable that we persist with the idea that such attacks are worthwhile. One reason, no doubt, is that we still find ourselves in the grip of the Injection Myth[3]. Another reason is that, when we want to influence someone, talking strikes us as the easiest and most obvious way to go about it, so we are reluctant to face the evidence which suggests that talking might actually be counter-productive (especially if we have some inkling of the fact that the strategies which are most likely to work are trickier to design and harder to put into effect than a bit of good old nagging, or a full frontal attack on someone's existing point of view).

But the most compelling reason why most of us persist with attacks on the cage is that we tend to think of attitudes and values (the bars in the cage) as being the *cause* of behaviour. People act a certain way, we think, because

[3] The 'Injection Myth' as described by the author in 'Why don't people listen' exposes the false belief that by simply telling something to people the meaning of the intended message will be automatically 'implanted' in their heads – as intended. Furthermore, the 'Injection Myth' implies that if at the first attempt we were not successful, then all we must do is to repeat the same all over again (Editors' note).

of the attitudes they hold. Therefore, the argument runs, if we want them to act in a different way, we must obviously get them to think differently first. Like so many other popular theories about communication, this one has some big holes in it. When we examine the relationship between attitudes and behaviour, we may be in for a surprise.

5.4 Attitudes and Behaviour: Which Causes Which ?

We notice many examples of people who act in ways which seem to be consistent with the attitudes and values they express. People who talk about ways to 'rort the system' tend to rort the system. People whose values seem to be materialistic tend to act as if material possessions are the most important thing in their lives. People who seem, in conversation, to have a caring attitude towards others tend to be involved in acts of charity and kindness. Such observations fuel the conviction that our attitudes and values shape our behaviour: from there, it is a short step to the conclusion that a *change* in our attitudes and values would *change* our behaviour.

But the relationship between attitudes and behaviour is not quite as straightforward as it may first appear to be. Think again about our cages and the way we construct them. From where do we obtain the raw material for those discoveries, learnings and decisions which form the bars of our cages? Where do our attitudes, beliefs, values and prejudices come from? (Wherever it is, surely that will also be the place where attitude *change* comes from.)

The answer, of course, is that our cages are built out of our experience. Our attitudes are the fruits of that experience. We construct the cage gradually, over a lifetime, *in response* to our experience of the world. (And, as we shall see later in this chapter, we modify the cage – change our attitudes – in response to new experience.)

So it actually makes more sense to think of attitudes as being the result of experience than to think of them as being the 'cause' of behaviour. That is an over-simplification, of course, because the relationship between attitudes and behaviour runs both ways.

Perhaps it would be fairer to say that our experience shapes our attitudes and our attitudes, in turn, shape our subsequent behaviour – pending *new* experience. This is precisely how we defined the operation of the cage[4]: we build the cage out of our experience and we then view the world through the bars of the cage. The cage is the mechanism for storing and using what we learn from our experience.

So the relationship between attitudes and behaviour is a two-way street, but the heaviest traffic flows *from* behaviour *to* attitudes, rather than in the opposite direction.

Because we form attitudes, values, opinions and beliefs in response to what happens to us, it is not surprising that, over time, there emerges a pattern of

[4] (Mackay , 1998, Chap.3)

broad consistency between what we think and how we act. Indeed the comfort of the cage depends upon achieving that kind of consistency.

But the consistency is by no means absolute: we often say one thing and do another. The American psychologist Leon Festinger has studied the relationship between attitudes and behaviour[5] and, in particular, focused his attention on the emotional problem created for us when we find that our attitudes and behaviour are out of kilter. According to Festinger, when we think one way but act another, we experience what he called 'cognitive dissonance' (that is, mental disharmony).

Being in a state of dissonance is uncomfortable for us, because of the tension created by a lack of consistency between the pattern of our behaviour and the pattern of our cage. We will want to reduce that tension by closing the gap between attitude and behaviour. The interesting question is*: how will we do it?*

If we believe that attitudes are the cause of behaviour, then we would expect to see people relieving the pressure created by a gap between attitudes and behaviour by modifying their behaviour so that it lines up with their attitudes. In reality, the more common response to 'cognitive dissonance' is *to modify our attitudes* so that they line up more comfortably with our behaviour.

For example, a boy might regard himself as being basically honest. He has incorporated into his cage the lessons of his parents which have been broadly supported by his own experience: he believes that 'honesty is the best policy' and that stealing is wrong. The boy is rather lonely at school and seems to have trouble being accepted by the other boys. Gradually, he finds acceptance within a particular group of boys who don't happen to share his view of the morality of stealing. After school, they frequently visit the local shops and engage in shoplifting as an act of bravado.

Desperate to be accepted by his new-found friends, the boy goes along with them and, under pressure from the other members of the group, he steals something from a shop. This makes him feel guilty: he is stressed by the tension between what he believes (shoplifting is stealing, and stealing is wrong) and what he has just done.

As time goes by and his shoplifting becomes more habitual, the boy finds that his guilt recedes and the tension between his attitudes and behaviour is reduced. This has not happened because he has modified his behaviour in the light of his attitudes; on the contrary, his attitudes have become more consistent with the new pattern of behaviour. Now, he is prepared to say to himself, 'Shoplifting is not really stealing... no one really suffers from it... it is just harmless fun'. He is in the process of changing his mind in response to new experience.

Another example of the urge to achieve consistency between attitudes and behaviour comes from Edgar Schein's classic study of American soldiers who

[5] (Festinger , 1957)

were taken prisoner during the Korean War[6]. Whereas the North Koreans favoured harsh punishment of prisoners to 'change their minds' in favour of communist ideology, soldiers who found themselves in prisoner-of war camps run by the Chinese received very different treatment – although the object was the same.

In attempting to convert Americans to the communist cause, the Chinese would typically ask their prisoners to make mildly anti-American or pro-Communist statements, which were so mild as to seem inconsequential. For example: 'The United States is not perfect.', or 'In a communist country, unemployment is not a problem.'

Once such mild statements had been made, the Chinese interrogators would then increase the pressure: having agreed that 'the United States is not perfect', a prisoner would be asked to list some of the ways in which America was not perfect... then to identify some of the 'problems' with life in America, and so on. He would be asked to sign his name to these relatively innocuous statements on the grounds that they were, after all, his own beliefs.

Later, the soldier might find that his signed statements were being broadcast to the entire camp, to other POW camps, as well as to American forces in South Korea. Now being identified as a 'collaborator', and realising that his statements had been made without any real coercion, a prisoner might modify his attitudes still further (in the pro-Communist direction) in order to line them up with his publicly exposed behaviour. Under such conditions, the prisoner might become even more cooperative with his captors. In studying this unexpected phenomenon, Edgar Schein reported that 'Only a few men were able to avoid collaboration altogether'. Clearly, under such conditions, the apparently harmless act of making the 'trivial' statements requested by the Chinese set up increasing tension between the attitudes and behaviour of American prisoners – tension which was relieved, not by changing their behaviour to be more resistant to their captors' overtures, but by changing their attitudes to make them feel more comfortable with what they had done.

People who experience religious conversion often do so in response to changes in their circumstances which cause them to perceive religious messages in a new light. Their attitudes change because their experience of the world has changed.

The popular cliché that 'travel broadens the mind' captures the same idea: when we travel and experience cultures different from our own, this experience has an illuminating effect on our attitudes and our cages are expanded accordingly. Even people with the most appalling racial prejudice may find that is it eroded by the experience of getting to know individuals who come from the racial group which was previously detested: new experience produces new attitudes.

The point about these examples is not simply that they demonstrate the primacy of experience over attitudes; they also demonstrate the fact that we

[6] (Schein et al. , 1961)

are much more likely to modify our cages in response to what we learn from our own experience, than in response to other people's attempts to change our minds through communication. We prefer to change our *own* minds in response to our own experience.

Women who have given birth to their first child often describe the experience of new motherhood as the most profound change in attitudes and values they have ever experienced. 'Nothing could have prepared me for this' is a typical response, as the new experience begins to reshape a mother's attitudes towards every aspect of her life – often sweeping aside attitudes towards housekeeping and parenting which had previously been firmly and determinedly held.

Life-threatening illnesses and life-changing experiences (marriage, divorce, bereavement, retirement) cause many people to reassess their values and priorities and to change their minds about things, which may always have seemed clear and unchangeable.

Of course, not all new experience has a cage-changing effect. One of the major themes of Mackay (1998) is that, even when we are confronted by new experience, we may filter perceptions of it, so that it remains consistent with the existing shape and structure of the cage. We *tend* to interpret new experience in the light of the lessons learned from prior experience.

So the relationship between attitudes and behaviour does not imply that new experience will always lead to new patterns of behaviour and, in turn, new attitudes. *Some* new experience will produce new attitudes; some won't. But everything we know about the cage suggests that people are much more likely to change their minds when they learn from a new experience, than when a message cajoles them or puts pressure on them to change. (There is still an important role for communication in the process of behaviour change and attitude change and we examine that role a little later in Section 5.7,'What is the role of communication in all this?')

A teenage boy is notoriously reluctant about personal hygiene. His mother is constantly at him to shower more frequently, to use a deodorant and to take more care with his appearance. He attends an all-boys school and travels on a school bus which only carries boys from his school.

Suddenly, it is announced that the bus is to be shared with girls from a nearby girls' school. After the first couple of weeks of the new arrangement, the boy's mother notices a dramatic improvement in his personal appearance and she begins to have great difficulty in extracting him from the bathroom. Have his attitudes changed? Who cares?

5.5 Is It Really People's 'Minds' that We Want to Change?

Our understanding of the complicated relationship between attitudes and behaviour raises an important question about the whole idea of changing people's minds: is it their 'minds' we want to change or is it really their behaviour that

we want to change? Do we only talk about 'changing people's minds' because we cling to the idea that an attitude change must precede a behaviour change? If it were true that new behaviour is likely to shape new attitudes (as long as we communicate appropriately with the person who is experiencing the new behaviour), mightn't this shift the focus of our interest?

This brings us to a point of significant convergence between two approaches to the process of personal influence which, at first glance, seem quite different. One approach says that if we acknowledge that the real focus of our attention is behaviour (not attitudes), then we might as well go straight to the behaviour itself, work on that, and let the attitudes take care of themselves.

The other approach says that although we may ultimately want to change someone's attitude, we recognise that the most effective way to do this is to provide the person with some new experience from which new conclusions might be drawn and, in turn, new attitudes formed.

Either way, whether our ultimate focus happens to be on attitude change or behaviour change, we are left with the probability that changing people's behaviour will be the most productive first step (in any case, it generally turns out to be easier to change people's behaviour than to change their minds: the fact that a change in behaviour may produce a corresponding change in attitude is something of a bonus).

From an ethical point of view, putting the emphasis on behaviour may be something of a relief. As long as we cling to the idea that the way to change people's behaviour is to get them first to think differently, we operate in the mysterious world of 'the mind'. As long as we believe that our task is to change people's attitudes by changing their minds, we are vulnerable to the appeal of subtle and devious techniques of psychological manipulation. It is a very short step from the belief that we have to change people's minds to the idea that it is acceptable to act like 'thought police' [7].

If we focus on the other person's behaviour, our motives will be rather more obvious than if we are lurking about in the shadowy world of 'attitudes' (and, if we are smart, we will consult with the other person about what we are doing and why we are doing it). Nothing is hidden; nothing is mysterious; we all know where we stand.

Yet, we persist with the idea that it is 'attitudes' or 'minds' which we wish to change. Parents talk about the need to change their children's attitudes towards table manners: what they usually mean is that they want to change the *manners themselves.* Road safety authorities talk about needing to change the attitudes of motorists who speed or who drink and drive: what they really want to change is *their behaviour.* Attitudes scarcely come into it.

When managers complain about employee attitudes, or teachers complain about the attitudes of 'today's students', they are generally expressing dis-

[7] The Political Correctness movement of the early 1990s attracted such hostile reactions precisely because it seemed to be trying to clean up our minds and our language... almost regardless of how we actually behaved.

pleasure with some aspect of the employees' or the pupils' behaviour. They may assume that the cause of the undesirable behaviour is a set of undesirable attitudes (and they may assume, therefore, that they need to work on the attitudes), but the real problem is the behaviour.

5.6 How Do We Change People's Behaviour?

Once we decide that it makes sense to focus on behaviour (even if we still want to think of it as a pathway to attitudes), the next question is: how do we go about changing someone's behaviour?

5.6.1 Change in the Environment

Any pattern of human behaviour – from driving a car to making love – is a result of the *interaction* between a person and the total environment in which that person is functioning: other people, the place, the situation, the circumstances. (The 'person' is an inextricable combination of unique inherited predispositions – genes – and the qualities and capabilities which have been shaped by all the influences on them over their lifetime.)

If we want to change the way someone behaves, therefore, we will have to change the nature of that interaction. To do that, we will have to make some change to things that are interacting. We will either have to change the person (always a tough assignment, and sometimes almost impossible in practice), or the person's perception of their environment (easier said than done, thanks to the cage), or the environment itself.

Most times, it is far easier to start by making some change to the environment – the situation, the system, the circumstances – so the person is given the opportunity to react and respond to some new experience.

This is not to suggest that humans are mere victims of either their genes or their environment, nor pawns in some cosmic game of deterministic chess. It is simply to acknowledge that *some* aspects of our behaviour (those things we do which we could do differently) are heavily influenced by our immediate circumstances, and may be changed by changes in those circumstances. Geography, climate, peer-group influences, money, comfort, convenience, equipment, illumination levels, architecture and urban design... these and many other factors in our social and physical environment are the kind of things which influence the way we behave. Of course, it is also true that there are basic human drives (like hunger, sex and the need for shelter) which have to be satisfied, but the way we satisfy those drives will be heavily conditioned by the environment we are in at any given time.

This is why people who are successful at influencing the behaviour of others generally start by looking for ways of changing the environment – the circumstances – in which they want people to act differently. They recognise that people tend to behave habitually (as they 'learn the ropes' associated with a

particular system or a particular set of circumstances), and those habits are unlikely to be broken unless the nature of the system or the environment itself is changed.

This is why road safety authorities have long since realised that propaganda on its own is less effective than propaganda which is designed to support direct intervention in the driving environment: the introduction of random breath testing units, for example, has a much more powerful effect on drivers' behaviour (and, ultimately, on their attitudes as well) than any amount of public education about drink-driving in the absence of environmental change.

This is also why factory managers have often found that productivity increases are more likely to follow modifications in the factory environment (better lighting, air-conditioning, dust filtration, sound dampening) than the endless sending of messages about improved productivity.

It is why so many marketing companies have found that in order to get consumers to try a new product, they must distribute free samples – or create other incentives to try it – rather than simply advertising it.

All these examples point to the general proposition that if we want people to behave differently, we must create the *conditions* under which it will be both easy and attractive for them to do so. Unless we modify the environment in some way, the people who are interacting with that environment are unlikely to behave differently, regardless of how persistently we might ask them to do so.

The chief executive of a major industrial company once remarked, during a period of instability associated with a program of decentralisation: 'When we have decentralised, we will have to begin centralising again. It is only when we are reorganising that people are flexible enough to adapt to new thinking.' That may have been an extreme approach (and it produced a few nervous breakdowns along the way), but he was on the right track: changing circumstances do increase the chance of more flexible patterns of behaviour.

A manager who wanted to introduce computer technology into the workplace found that the employees concerned were very resistant to the idea. Instead of simply trying to persuade them to his point view through communication, he offered to install a number of computers in the staff canteen, loaded with a range of intriguing games. There was nothing devious about the manager's approach: he explained exactly what he was doing and why he was doing it, and made it clear that he wanted the employees to react to his proposals on the basis of experience, rather than ignorance.

With a small amount in instruction, employees began to play computer games during their lunch break and, in the process, to familiarize themselves with computer operations. In the end, their resistance to the idea of using computers was broken down by the experience of actually operating the new machines.

You want to get some to act differently? It is very unlikely to happen unless you do something to change the system or the environment in which they are operating.

A mother wants to encourage the members of her family to put dirty clothes in the laundry basket rather than strewing them all over their bedroom floors. She devises a system which can be easily explained but which represents a significant change in the running of the household. 'From now on', she says, 'I will only wash clothes that are in the laundry basket. I don't mind if you leave things on your floor, but I am not going to operate in any other way. What you put in the basket will be cheerfully washed. What you leave out will be unwashed. Simple as that. You do your part, and I'll do mine'. At first, the family might think that this is just another idle threat and may not appreciate that the system has, indeed, changed. It is crucial that she should adopt the system with the same rigidity as she expects of the others. She mustn't weaken, even if clothes pile up, unwashed. Sooner or later, everyone will decide that it is nicer to have clean clothes...especially when all that is required is that dirty clothes should be dropped into the magic basket instead of onto the floor.

The key to attitude change is behaviour change, and the key to behaviour change is to change the system or the circumstances, after fully explaining the change and the reasons for it. It is not a sure-fire recipe, but it beats nagging, or any other strategy which relies on communication alone.

So, although we can never make absolutely general statements about the ways in which attitudes and behaviour affect each other, and although there is no 'magic formula' for changing people's behaviour, the Seventh Law of Human Communication[8] describes our best chance of success:

People are more likely to change in response to a combination of new experience and communication, than in response to communication alone.

Some of the most extreme methods of changing people's minds by modifying their behaviour were documented in the landmark book about brainwashing, *Battle for the Mind*, by William Sargant[9]. Sargant pointed out that the process of indoctrinating political prisoners depended on first disrupting their environment to such an extent that they could no longer rely on their established framework of attitudes, values and beliefs – nor on the support of their peers.

Disorientation of prisoners has become an established strategy for interrogation by secret police in totalitarian regimes. Arrests in the dead of night; withholding of information about the nature of a charge against the prisoner; isolating the prisoner; keeping the prisoner in permanent light or permanent darkness (or altering those states unpredictably); alternately depriving the prisoner of food and then offering lavish meals; giving outlandish information which, though the prisoner would be disinclined to believe it, could not be

[8] (Mackay , 1998)
[9] (Sargant , 1963)

checked against any other source. In these and other ways (including physical torture) interrogators can so disrupt the normal rhythm and pattern of a prisoner's life that they become open to suggestion[10].

William Sargant noted, however, that even attitudes which change under such harsh conditions are likely to change again when prisoners return to their normal environment. Though some people never recover from the destruction of the cage wrought by such savage treatment, others rebuild their cages in the light of subsequent experience. (The American prisoners of war in North Korea, for instance, though heavily indoctrinated with pro-Communist messages, typically reverted to their original anti-Communist attitudes once they were repatriated.)

Sargant documented the extraordinary forms of environmental manipulation which have been used by various religious sects and fanatics in order to induce states of near-collapse in people who are targeted for 'conversion'. Everything from the snake-handling of some American religious groups to the intense rhythmic drum-beating of voodoo cults was designed to create an atmosphere so removed from 'normal' that people caught up in it would abandon their usual perspectives.

But these are extreme cases. More benignly, the founder of the Salvation Army, William Booth, recognised that a destitute soul is more likely to listen sympathetically to the preaching of the Christian gospel on a full stomach: soup kitchens therefore became an integral part of the early ministry of the Salvation Army.

5.6.2 A Case Study

"Remember when Margaret was trying to get Michael to wipe his feet" (Mackay , 1998, Chap.2)?

Like most parents, Margaret's first resort was to talking: it always seems easier to talk than to take more positive (and effective) action. But suppose she locks the door when Michael is outside and then when he tries to get in, tells him that she will unlock the door after he has wiped his feet.

This strategy involves an effective combination of a 'system change' and communication: by locking the door, Margaret has created a new set of conditions under which her message will have immediate relevance to Michael's situation: he can't get in until he wipes his feet.

Of course, doing that once probably won't solve the problem. But if she does it three or four times, a new pattern of behaviour will begin to be established (especially if Margaret reinforces the new behaviour with plenty of encouragement and praise).

[10] More recent experience in the Middle East has shown that under conditions of severe disorientation, prisoners can actually form an emotional bond with their captors because their normal frame of reference is shattered and they come to rely on the only human network to which they feel they still belong.

That suggestion may well provoke cries of protest from some parents, who will say, 'When have I got time to go locking the door three or four times, just to train him to wipe his feet? Tell me how I can get him to do what he is told... that's what I want to know.'

But this *is* a way of getting him to do what he is told: this approach certainly involves more short-term effort on the parent's part, but a much higher probability of permanent success. It will work better than nagging, and it is also preferable to the other obvious alternative: punishment. (Punishment may have its place in child-rearing practices, but it is usually better to look for some positive strategies for change before we resort to the negative approach – which is often simply an admission of our own failure to achieve the result we wanted.)

Which is best: to go to the trouble of locking the door three or four times to establish the new pattern, or to keep telling Michael – perhaps dozens of times – to do something which it is easier not to do? The truth is that when this pattern of behaviour has been explained, established and reinforced, Michael will be more likely to respond to similar messages in the future. The change in his behaviour is likely to produce a corresponding shift in his attitudes.

Hitler needed his rallies; Wesley needed his hymns: their messages alone were not enough.

It was Australians' experience of actually using Bankcard which changed their attitudes to credit: they did not begin using Bankcard because they had already changed their minds about credit.

Why does canned laughter on the soundtrack of a TV program encourage us to laugh at what is going on the screen? TV program producers know that viewers are less likely to laugh when they are sitting in isolation: modifying their environment by creating the illusion of other people laughing frequently produces the desired effect.

'Plant a kiss on the Cole-face. How nice to see you.' Margaret's father, Cole, has made one of his rare visits to see her. Michael, his only grandson, is the apple of his eye. He invariably refers to Kelly as his surprising little step-grandchild, a label which Kelly has learned to hate because she perceives – correctly – that Cole is rather uncomfortable about Margaret's status as Bill's second wife.. the idea of step-relationships vaguely bothers him. Cole himself has been married three times but, as he himself is fond of saying, 'All my children are my own' – a line which infuriates both Margaret and Kelly every time they hear it. He is currently unmarried. He calls it 'floating' and Margaret knows that he is floating in some pretty fast currents. Cole is the host of a long-running radio program, 'Metaphor', in which he interviews people about significant events and symbols in their lives. His audience is tiny, leading to another of his slogans, 'Never mind the width, feel the quality'. 'Metaphor' is under constant threat of being dropped, but Cole is passionate about it and talks of little else. Bill and Michael are at the park. Kelly is at netball. Margaret is about to have a rest when Cole arrives on the doorstep unexpectedly. She makes tea and they sit on the back verandah. 'You simply wouldn't believe

what the latest thing is. Security. We've all got these magic pass things which we have to insert into various slots and things to make our way about the building. Unbelievable. You simply can't get in without a pass. And we wear these all the time.' He holds up an identity card, which is hanging on a chain around his neck. 'I find it easier to leave it on all the time. Easy thing to forget. Like my glasses: keep them on and you know where they are. Though that's not entirely foolproof either...I have found myself searching for them when they were on my nose the whole time. Anyway, where was I?' 'Security. The magic cards.' 'Ah. Yes. The thing was, there were a lot of thefts going on. Quite extraordinary in a place like ours. I blame the ethnics of course – there are a lot of very strange people around the office since we've got into some of this foreign language stuff on the air. But of course you can't pin anything on anyone. And you can't say a word against any of these people, you know. The thought police are everywhere.' 'So has it worked?' 'Come again, old thing?' 'The security. Has it worked? Have the thefts stopped? 'Well, that's the extraordinary thing. Yes, they have. But there was more to it than that, of course. There are a lot of very important people coming and going in that building every day...to say nothing of all our tapes and archival material. There are years and years of 'Metaphor' tapes there, for a start. So there was a sort of point to it. But yes, the thieving has gone away. So far, anyway.' 'So it wasn't the ethnics after all.' 'Well, who knows. But yes, I see what you're driving at. Must have been people coming in off the street.' 'And the program is surviving. I'm afraid I don't get to hear it all that often... ' 'Oh surviving. Absolutely. New lease of life. But the place isn't the same. That's another thing about this security bizzo. You solve one problem. You create another. People don't move around as freely as they once did. Chewing the fat. That type of thing. It's more a case of heads down and get on with it. Funny thing. And I'll tell you something else. We're all a bit more organised than we used to be. I find I get through a lot more in a day. You don't mislay stuff like used to ... at least I don't. There's not the same amount of bumf drifting about. Bits of paper. Blank tapes. Things like that. It's more under control somehow. So it's an ill wind...' Margaret finds herself thinking that there was probably never any theft at all. If Cole can't remember where his glasses are, he has probably mislaid an entire forest of files, over the years. But it makes a good story. Everything Cole talks about makes a good story. 'Will you stay for dinner, Dad?' 'Look, I can't. Got something on, I'm afraid. I should pop in more often. Have I missed the kids?' On cue, Kelly comes through the door, just home from netball. 'Ah. My little step-grandchild. Goodness me. Not so little. Come and plant a kiss on the Cole-face. I'm just going. So nice to see you both.' When Cole has gone, Margaret reflects on the striking similarity between the new security system at his office and her campaign to get Michael to wipe his feet. Locked doors have their place in the scheme of things. And any system that can get Cole organised and focused on what he is doing must have something going for it. The security aspect would be a bonus.

5.6.3 Benefits and Risks of Environmental Change

Every manager knows that the layout of an office – and the nature of the office systems – will have a direct effect on the work practices and productivity of the people who work there. Noise level, territorial space, privacy, visual distractions ... all such factors influence the way in which people work and *changes* in a working environment can produce dramatic improvements or deterioration in the morale of the workers and in the quality of what they do. In other words, when our circumstances change, we may be destabilised by the change and we may have to revise our existing preconceptions and prejudices. Our cages are very strong and can withstand very considerable attacks upon them. But even the strongest cages may be disturbed by a significant set of new experiences, which produce new patterns of behaviour from which new attitudes may evolve.

That is the kind of situation – applied to an entire society – described in *Reinventing Australia*[11]. When a community is being subjected to widespread and sustained social, cultural, economic and technological change, feelings of uncertainty and anxiety are likely to occur. In turn, that feeling of instability creates a desire for new certainties and new sources of security. At a time of social dislocation, for example, people will often turn to extremist beliefs – in religion, in science, in feminism, in economics or even in astrology. When the world continues to be unpredictable, we have little alternative but to modify our cages in response to the prolonged instability of living through a period of discontinuous change.

Such a period – in the life of a community or of an individual – creates a field day for manipulators of every kind. When cages are fragile, people are vulnerable to new messages – especially those which offer the promise of a new sense of security. In the same way as we might try to change people's behaviour by deliberately changing something in their environment, so a spontaneous change in their environment may make them more willing to change.

But the opposite reaction may also occur. A boom in nostalgia often follows a period of prolonged instability, as people yearn for a return to 'the good old days' or 'traditional values' in an attempt to re-establish the comfort and security of the cage. Instability by no means guarantees an openness to new ideas: it may just as easily result in a reactionary retreat to an idealised past.

5.7 What is the Role of Communication in all This ?

It is not simply the change in the environment that does the job, on its own: Margaret's locking the door needed to be supported by an appropriate explanation. The best long-term results come from creating new experience by creating changed circumstances while, at the same time, explaining what

[11] (Mackay , 1993)

we are doing and what we hope to achieve. Road safety authorities do not simply install random breath-testing units around the countryside: they alert us to the fact that they are going to do so, and they explain the purpose of the strategy.

A marketing organisation does not simply distribute free-samples of its product: it also distributes information about the product and follows that with an advertising campaign designed to reinforce the new attitudes which will be emerging from the new experience of using the new product.

This is why the Seventh Law of Communication states that people are more likely to change in response to a *combination* of new experiences and communication, than in response to communication alone. (Even the terrorist does not leave it to our imagination to work out what outcome he wants: the terror is combined with clear messages about the desired result).

When we are trying to change someone's behaviour through a change in their circumstances, there are two roles for communication:

- We need to send messages to the other person which will act as a signpost, identifying and explaining the changes we are proposing to make;
- We need to use communication as support and encouragement for people who are adapting to the new circumstances.

5.8 What if We Can't Change the Environment?

The Seventh Law of Human Communication emphasises that it is the *combination* of new experience and communication which is most likely to produce an enduring change in people's behaviour (and, in turn, in their attitudes). But there will be many occasions when we may want to influence other people without being able to make any of the changes to their environment which might exert a direct influence on their behaviour.

A doctor trying to modify a patient's diet, for example, has no direct control over the environment in which the patient buys or eats food.

In such cases, communication may be the only available tool. This is not a hopeless situation, but it is certainly more difficult than one in which we can introduce some environmental change as part of our strategy.

If you are trying to influence another person solely through the process of communication, there are three things to remember.

5.8.1 Use the Existing Cage

Base your message on some existing part of the other person's cage. Try to reinforce an existing attitude and relate it to your message. Let the other person see how the behaviour you want is consistent with what they already believe.

Range Rover created an effective advertising campaign based on the slogan, 'Write your own story'. They were not persuading people to change their attitudes: they were persuading people who *already dreamed of escape* to focus those dreams on Range Rover. Their purpose was to change people's behaviour by tapping into an existing attitude.

5.8.2 Change Behaviour Rather than Attitude

Focus on the behaviour you want, rather than dwelling on the subject of 'attitudes'. People are much less likely to change in response to a request for them to change their 'minds', than in response to a request to modify their behaviour – especially if it is only a relatively small modification.
The parent who asks the messy teenager to' get your act together ... clean up your attitude... I don't know how you can live in this mess' is very unlikely to produce a positive result. But the parent who concentrates on one aspect of the child's behaviour is more likely to succeed: 'Can we find a way of organising the books you are going to need for school tomorrow, so you won't leave any behind? Don't worry about the rest... Let's just work out a system so you can have tomorrow's books organised and ready.' Such a request focuses on a specific aspect of behaviour and bypasses any general problem associated with an underlying 'attitude'.

5.8.3 Small Step(s)

Seek a small step (or a series of small steps, over time) rather than a giant leap. Erosion is more effective than explosion, especially when each small step in a gradual process can be reinforced through the use of encouraging and supportive messages. (This is precisely the technique used by commercial organisations that create advertising messages to reinforce the attitudes of people who are buying their product in the hope of encouraging them to keep doing so.) Even the tiniest steps in the right direction should be reinforced through constant encouragement.
The rural education officer who is trying to persuade a farmer to experiment with heavily increased doses of superphosphate is unlikely to succeed if he attacks the farmer's attitudes: 'I don't know why you are always so resistant to change.' But if he switches his focus from the farmer's attitudes to his behaviour, he may still fail if he asks for a behaviour change which is too radical: 'Why don't you double the dose on your whole property and just see what it does to your yield next season?' He is more likely to succeed if, without raising the question of underlying attitudes, he asks the farmer to double the dose on a small, experimental paddock: 'Let's see what happens next season if you just double the dose on your small back paddock. Nothing much will be lost but it will be a very useful experiment to see if this might work on your type of country.'

5.9 Consultation: The Key to Managing Change

Some of the material in this Chapter might appear to be suggesting that in order to influence other people's attitudes and behaviour, all you have to do is manipulate their environment in some way, provide a bit of encouragement and reinforcement, than sit back and watch them react, like so many rats in a laboratory.

That kind of approach would raise serious ethical questions. The underlying message of this Chapter is that communication is most likely to be successful when it takes place in the context of personal relationships which have some inherent integrity: relationships based on a spirit of mutual respect, trust and reciprocal obligation. When it comes to influencing other people's behaviour, that message still applies.

The process of personal influence should take place in a morally sensitive climate and, indeed, is more likely to succeed when that is the case. We should be prepared to be quite frank and open about what we are doing. We should explain our motives and our goals. We should communicate as openly a we can about the changes we are making to the other person's environment.'

Above all, we should consult with the people we are trying to influence *before* we design any strategies for getting things done differently.

At first glance, this may seem nonsensical. How, you may ask, are you going to influence someone to act differently if you make it transparently obvious to them that that is what you are doing? The answer is that you are even less likely to succeed if you attempt some kind of devious manipulation: there are no foolproof strategies for influencing other people, and the most effective strategies depend on achieving some degree of cooperation. For example, think again about Margaret trying to get Michael to wipe his feet. Simply asking him to wipe his feet has not worked because there is no pressure on him – apart from nagging – to take the message seriously enough to act on it. But when Margaret decides that she is going to develop a strategy to influence his behaviour in the direction she wants, she should certainly discuss it with him and, in consultation with him, explain what she is going to do and how she is going to do it. 'I know you can't help forgetting about wiping your feet. But it is important for you to learn to remember because none of us like having a dirty floor to walk on. So I am going to help you remember by locking the door when you are outside. It will be a bit annoying until you get into the habit, but at least I won't get angry with you.'

Michael is only seven years old, after all, and that degree of consultation would be easy to manage. Were he older, the consultation could be even more wide-ranging, so that he might be encouraged to make his own suggestions about how to achieve Margaret's objective.

In the area of workplace change, the same principle applies. If we attempt to implement change without consulting those who are going to be affected by the change, they are likely to be more resistant to it than might otherwise be the case. Remember the factory manager who installed some computers

programmed for games in the staff canteen as a means of letting the staff get used to computer operations in a non-threatening environment? His motives for putting the computers in the canteen were transparent: indeed, he explained to the staff that before any decisions were made about switching to a computer system, he wanted them to have opportunity to 'fiddle about' with a few computers, just to see how they felt. No obligation; no commitment; just try it, and see how it goes.

Environmental modification without appropriate consultation and explanation amounts to manipulation or even exploitation. Government authorities who institute changes, for example, in the management of traffic (through such initiatives as speed cameras or random breath testing) need to go to great lengths to consult with the community and to explain to the community the nature of the problem and the reason why this particular solution is being tried. In the absence of such consultation and explanation, those initiatives would, quite correctly, be interpreted by the community as an example of 'Big Brother' at work.

People who are brought into the process of decision making about changes which affect them are much more likely to accept that the changes are necessary, and are much more likely to participate in experimental change programs to which they have contributed even if they are lukewarm about them.

In *Organisational Change by Choice*[12], Dexter Dunphy urges managers to adopt an open and consultative style so that employees themselves will become involved in the design of strategies for change:

Some managers may see this as a loss of control. It ***is*** *a redistribution of power. Our experience is, however, that systems of mutual influence operate more effectively than situations of one-way influence. It is true that the managers, owners or employers may no longer get their own way. But they seldom do so now and attempts on their part to apply coercion often result in an equally strong or more powerful countervailing force that negates their power.*

None of this is to suggest that people in positions of authority (in management, or in a family) should abdicate their authority in favour of some wishy-washy process of consensual decision-making by the group. Group decisions tend to be notoriously bad decisions (because no single person accepts full responsibility for them). But people in positions of authority make the best decision when those decisions are *informed* by a process of consultation with those who will be affected by them.

The Eighth Law of Human Communication makes precisely that point:

People are more likely to support a change which affects them if they are consulted before the change is made.

[12] (Dunphy , 1981)

Why don't people listen? Sometimes it's because they weren't consulted about a change which is going to affect them and so, when we tell them about it, they say, 'First I've heard of it' and that's the end of that.

Being consulted about a decision is not the same as making the decision. Consultation is the process of involving all those who will be influenced by a change in the development of the strategies which will bring about that change. But, as Dexter Dunphy points out, the legitimacy and effectiveness of consultation depends on all of us being prepared to be influenced by it.

David, the branch manager for Croydon Bridge, is on the phone to Bill. "I've got a pretty unhappy bunch of people up here. They've just seen Christina's missive about the new forms for the monthly returns and they're up in arms, to be frank.' 'You don't have to tell me, mate. You're the third call this morning. I didn't know anything at all about this until I saw the new forms last night, so we're all in the same boat. Which is no help at all, I realise.' 'What happened to all the hot air about consultation? Out of the window, by the looks. My people reckon this system simply won't work. They can see about ten holes in it, straight off, and they just can't understand why no one spoke to them before this was designed. 'Reminds me of the classic about the printing shop. Did you ever hear about that one? Monumental cock-up. Some draftsman drew the plan for a new collating machine which was to go into the printery, but he never bothered to go and look at the place for himself. Thought it was all routine. Just drew it up off the existing plans he had on file. No one told him the place had been renovated since those original plans were drawn, and there was a bloody great pillar right in the middle of the spot where the new machine was supposed to be installed. There was one heck of a row about that. Same problem. Lack of consultation. The blokes in the printery weren't amused at all. As they said, one quick phone call would have solved the whole thing.' 'That's exactly what the people up here are saying. They run the system. They know the ropes. You'd think it would be pretty fundamental to bring them in at the beginning of a process like this. You can imagine what it's done to morale. Rock bottom again. Part of the problem is that we've keyed everyone up to think that things are going to be different.' 'I know what you mean. Anyway, a spot of interesting news from home. Old King Cole came around to see Marg last weekend. She tells me he's belly-aching about their new security system. It seems he can't lose anything any more ... so I suppose he's afraid of being found out. There is a God, after all. Anyway, I'll look into this monthly return thing and get back to you. In the meantime, you'd better tell your troops to ignore the new forms and just carry on regardless.' 'That's exactly what they've decided to do.'

References

Dunphy, D. (1981) Organisational Change by Choice. Sydney, New York : McGraw-Hill.

Festinger, L. (1957) A Theory of Cognitive Dissonance. Stanford, CA: Stanford University Press.

Sargant, W.W. (1963) Battle for the Mind. A physiology of conversion and brain-washing. (Revised by the author). London : Pan Books (first publication: London : Heinemann 1957).

Schein, E.H., Schneier, I., Barker, C.H. (1961) Coercive Persuasion. A socio-psychological analysis of the "brainwashing" of American civilian prisoners by the Chinese Communists. New York : W. W. Norton & Co.

Mackay, H. (1998) The good listener : better relationships through better communication Sydney : Macmillan.

Mackay, H. (1993) Reinventing Australia Sydney NSW : Angus & Robertson.

6

CAPABILITY IMPROVEMENT

Ted Goranson

Advanced Enterprise Research Office, Old Dominion University - Virginia (USA)

6.1 Introduction

The notion of enterprise integration revolves around the central idea of enterprise modelling. Enterprises can be 'integrated' at the level of basic services, at the level of applications or at the level of models. Market forces drive to the second of these, because a robust market exists for applications (and related tools); application integration frameworks are therefore well supported by the marketplace. At the time of writing, two major camps of application frameworks are evolving:

- one devised by Microsoft, and
- one by everyone else, based on Java and UNIX.

There is such strength behind trends associated with application integration, that it gets confused with enterprise integration, with resulting confusion about underlying methods and goals.
The idea of deriving models of *elements* of an enterprise became practical in the 1960s. Initial motives were to make a process comprehensible by making it explicit; once past that barrier, managers and engineers could understand and analyze it in various ways, usually as part of a goal-driven improvement initiative. However, the notion of making processes discretely explicit has driven a quite revolution in management science. It is now a standard assumption that processes can be understood by managers all the way up to external financial analysts, and that all essential factors in a process can be engineered. So the basic notion of individual process modelling has enabled activity based costing, knowledge management, enterprise resource planning, product data management and many others, depending on the motives and languages of the interested parties.
Enterprise modelling first became practical in the 1990s. The basic notion of enterprise modelling is to create an integrating framework into which all component models can fit, so that interdependencies can be made explicit and

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

a combined model of sorts can be seen of the entire enterprise and be analysed and optimised in much the same way as the component models. Most of the other Chapters in this book describe different approaches to this framework notion, but the notion of combination or aggregation or integration at the model level is the constant. This movement, overwhelmingly supported by corporate users, grew from two needs:

- Optimising processes and combining them almost never results in an optimised system. There are several sub-optimising drivers, mostly deriving from local metrics. By combining everything into a 'big picture' model, one can optimise the enterprise against some sort of goal – usually some sort of near or longer term profit – and then project the changes to component processes. There is enough experience to clearly observe that many process changes that result from this vision are non-intuitive and would never have been seen without the big picture.
- A typical enterprise has many types of managers collaborating in some sort of equilibrium. Observers of management history note that since automation, the functions (marketing, engineering, etc.) have grown apart as they have evolved information pools that they own. These managers often compete with each other – often implicitly – as much as they do with bona fide competitors. The enterprise model not only provides a framework in which process dependencies link up, but also (usually) includes frameworks for user 'views' – financial, operational, strategic, etc – and entities they define: processes, resources, organisation, information and such. This framework provides a rationalising balance that supports reasoned trade-offs between management powers within the organisation and in the best frameworks give perspective for sub-optimising functional metrics. For instance, some financial manager may focus on optimising resources and be balanced by someone focused on gaining market share. The enterprise model framework provides (in theory) a way of coherently sub-optimising these metrics in much the same way that processes are sub-optimised.

6.2 Problems and a New Role for Enterprise Modelling

Enterprise modelling originated in the operational side of the enterprise and traditionally it was focused on that part of the enterprise with which operational managers concern themselves. This meant that the strategic planning of the enterprise, the marketing strategy, the design of a financial strategy to support the business model (indeed refinement of the business model itself), and major parts of the product design (if applicable) were presumed to have taken place before the enterprise modelling process began. In fact, the results of these operations were structurally excluded because they defined boundary conditions for the framework.

Certain assumptions were also usually made about how stable the decisions were from those preparatory decisions: they were presumed to be fairly stable.

In this context, enterprise models were build with the intent of 'engineering' the enterprise to optimally support these presumptions. It would then operate in a fairly stable mode for a long time. After some time, prompted by external changes and/or new insights or improvements in 'internal' processes, the enterprise would be 're-engineered' in order to operate in the new configuration for another long, stable period.
There are all sorts of disadvantages to this use of the powerful (but usually costly) idea of enterprise modelling. The major problems are that high risk processes are excluded, and dynamism within the enterprise is not well supported in response to new conditions or insights. For a few years, the enterprise integration community was stymied by these limits, which were not immediately self-correcting from the market or from other managers because of the forces already noted.
But we are now well into the evolution of enterprise integration via enterprise modelling into a new, 'second generation'. With this new generation of frameworks, strategic and key business infrastructure *is* incorporated into the enterprise modelling framework. This will allow the 'balance' of operational decisions to be closely tied to strategic business goals. Evolution in another dimension will allow continuous engineered adjustment to the enterprise in response to dynamic·conditions, rather than waiting for big 'batch' changes.
This Chapter addresses these two dimensions together, its focus being on the business mechanics and modelling challenges involved in evolving to this next generation of frameworks. The Chapter does not address pay-offs, enabling technologies or framework market forces – as they are addressed by other chapters – except to briefly note:

- The pay-off from agile, coherent system-wide management systems is potentially enormous. Some inkling of the recognised demand is the current popularity of 'balanced scorecards' which emulate next generation model frameworks through the relatively crude mechanism of metric frameworks. Metric frameworks can only be used retrospectively for reporting, but not for control or active management, because metrics are results of processes, not processes themselves. Their popularity, however, indicates that there is a hunger for immediate visibility and linkage with the whole system;
- The technologies and business methods required for second generation systems will almost certainly be ontology-based, use agents in advanced ways and model uncertainty. This last comment is probably the greatest technical challenge, the solutions of which at the time of this writing are just emerging;
- The market forces for enterprise integration frameworks and modelling support will probably not emerge directly from the (relatively small) market for operations managers. Instead, it will probably break through as a next generation of a merger of knowledge management and activity-based accounting and will supplant or subsume current enterprise resource and supply chain management systems. That is because markets are no longer

defined by underlying technologies, nor effects in the enterprise, but by the types of managers targeted in the enterprise.

6.3 Agile Virtual Enterprise and Fluid Supply Chains

The goal of second generation enterprise integration systems is to create adaptable systems that understand their environment, adapt themselves in some way to address opportunities and deal with threats, and once committed, continue to organically improve in a deeply self aware manner. Perhaps that organic change may trigger an exit from the market, or migration to an entirely different business model. And to do all of this in a deliberate, managed way with auditability and external visibility so that investors and customers have confidence in the whole system.

That is a rather tall order, and is radically different from the way things were done with first generation systems. The vision of how this revolution will play out depends on the perspective. There are two such perspectives: agile virtual enterprises and fluid supply chains.

The fluid supply chain idea is less revolutionary, and pegged to the fact that large user firms dominate the future of infrastructure systems. In fact, since manufacturing has a very demanding requirement for coordination of state, the large manufacturing enterprise drives the market for service and financial infrastructure as well; especially notable are the interactions among these sectors.

At any rate, these firms have a vested interest in centralisation – in fact, their survival depends on taking advantage of the next generation of management systems to increase concentration of key elements of activity 'at the top' of a supply or value chain. Next generation enterprise integration systems will allow the ability to quickly comprehend supplier / partner processes, understand cost / benefit issues of change, and effect those changes resulting in at least an equally robust management infrastructure. At least some measure of simulation will be supported; in this case, 'what-if' enterprise models can be constructed to explore possibilities of changes and responses to threats and opportunities (e.g. via scenario simulation), supported by so-called 'fact-based' decision making.

The result will be organisations that superficially look like they do today, but with greatly enhanced ability to leverage intellectual property, a high level of managed dynamism, and a further exploitation of niches that previously were too hard to access. So-called 'lean' processes will be delegated to lower levels of the organisation (and third-world operations) and the watchword will be *value-driven processes.* Supply chains will be less permanent, with internal infrastructures less shaped by their primes, while providing greater visibility and potential control. Capital will flow more freely, because decisions will be more auditable and rational. Whatever buzz-phrase succeeds 'the learning organisation' (and means the same thing), it will be popular.

A complementary, parallel vision is already emerging, which is a somewhat more radical business model: the *virtual enterprise.* A simple notion of a virtual enterprise is a group of small firms that semi-permanently ally to market themselves collectively. A much more interesting model is the 'opportunity driven' virtual enterprise. In this case, the enterprise is actually created in response to the opportunity. Presumably, this kind of business model will be appropriate for opportunities that need to the 'probed,' are profitable niches or temporal windows of opportunity that big, slow enterprises cannot address, or businesses where risk is high because conditions are expected to change or unexpected threats are common. In this latter case, the advantages of the virtual enterprise are that it can continually improve in response to changes in the environment.

The more advanced case is often called an *agile* virtual enterprise, and can be seen as a continuum in the evolution of business models in the way that capital and resources are linked. With the large, vertically integrated enterprise, the management of capital is intrinsically linked to the management of production. Hence, investment in a product means investment in the production assets, including worker and corporate knowledge. The result is that capital clock speed is linked to the investment life of the asset, usually a very long time. As this type of enterprise strives to become more lean and profitable, it must become less flexible and able to continuously improve.

The first major evolution was to migrate much of the deliberate management to the 'invisible hand' of market forces. Thus, there is the supply chain where the production can be managed very much like in the vertically integrated case, but the management of capital has been somewhat decoupled from production. Each supplier has a business boundary around it, within which they have to develop, maintain and improve their resources independently of the other suppliers. Because each supplier exists in a competitive environment, market forces drive them towards improved capabilities. The risk / reward mechanism over time encourages this improvement. Market forces handle much of the capital management, while management of production in this first evolution remains with the prime.

But in that case, there still is a tight link between capital management in the sense that investments for large improvements are explicitly underwritten by the prime in guarantees tied to collaborative metrics. The virtual enterprise takes the evolution one step further by making improvements in production: a distributed process controlled by market forces.

In both the virtual enterprise- and supply chain cases, the central dynamic is the notion of *capability improvement.* 'Capability' includes the tactical values of faster, cheaper and better, but it also addresses notions related to greater likely profit in the future: enhanced stock value (and availability of practical capital), increased access to markets (and other stakeholder goodwill), improved internal knowledge and skills portfolio, greater access to partners, and reduced high level risks.

6.4 Dimensions of Capability Improvement

As with all good business models, agile virtual enterprises and fluid supply chains are not dependent on technology for their value. Instead, they could be influenced by four notions which are suggested to be intrinsic positives that have not fully manifested because of some infrastructure barrier that will be mitigated by second generation capability improvement infrastructure. Because they are good business practices, one already sees instances ogf these factors in areas not so constrained by those barriers. These are: value features, federation, agency, and introspection.

6.4.1 Dimension: Value Features

In order for a system to see itself and continuously improve, it needs to have something more than a framework into which models are interrelated. The system needs to have some *meaningful token* which can be exchanged between model domains. The idea behind an enterprise model framework is that diversity among different types of models and elements is recognised and accommodated. Interdependencies among those processes represented are sufficient to do useful top-down analyses. But continuous improvement is largely dependent on *bottom-up* views: a process owner wants to see how the process fits in the big picture and what can be done to improve the enterprise.
The problem is that the process features will be locked into a domain and the metrics particular to that domain. For example, if in the manufacturing domain, particular concern will be with process features such as machining tolerances and throughput. In a particularly well managed and modelled enterprise, there will be some quantitative mapping to a set of process features that matter – to be able to determine that the attention spent on very fine machine tolerances can, and should be relaxed in order to increase throughput, or vice-versa.
If in product design, a different set of features would shape decision making, e.g. features having to do with the utility of the product. In this case, to be able to determine that a slight additional cost in materials would result in a far superior product. The essential problem in enterprise integration for the purposes of continuous improvement is that an enterprise consists of many such 'feature types', and that there is no coherent way to map from one to the other. How can the process and product owners relate their speculations about what might be the best alternative, if they cannot map one feature set to another?
An extremely reduced example: a process owner knows that they currently make widgets in three colours, but because of new information (perhaps a rise in the cost of raw materials) it will now be more costly to change colours. At the same time, new materials would lower the cost if a new machine were used. The implications of using the new machine and restricting colours needs to be accessed. In an unintegrated enterprise, the parties involved (even though

it is not at all clear who they all are) need to be assembled and a reduction in the dialog to dollars has to be effected because that is the only common language that is understood by all parties. Even if successful, this process will not be able to trigger change, but instead just send information up the chain of command for a 'real' decision and the triggering of change.

In a capability improvement-friendly integrated enterprise, the models are not the only level that is integrated; the *features* of key model domains are abstracted and integrated. There are in this case 'value features' that represent a rich, qualitative model of the enterprise's goals. These are not merely customer values; they include and subsume them because the enterprise may - almost certainly will - have strategic goals beyond greater satisfaction of a specific customer set. An example of this is that the enterprise may want to increase internal skills and resources for a strategic purpose (say, financial analyst economic value-added metrics), so process changes are driven by those features rather than direct customer pull. Or, alternatively, the customers may not yet value the features of the next generation of product.

All this is to say that when integrating, the best approach is not to have a 'customer-driven' set of enterprise features. Instead, when developing a set of enterprise value features, the normal approach is to develop features that directly address strategic goals, then map these into product / customer / service features, then (if a manufacturing enterprise) to physical features and then to process features. One cannot have meaningful enterprise integration for capability improvement unless some mechanism for this exists. Usually, it is some top executive's intuition, but it needs to be made explicit and visible to all. So, when the process owner evaluates the colour and machine alternatives, a clear auditable logic chain is made directly to the enterprise's strategic goals and the implications of change are visible.

Below, the abilities are added to effect well founded change, and to evaluate dependent changes among many processes.

Some comments on value features are probably useful. Enterprise integration for the visibility of process and resource dependencies is relatively easy. Integration at the higher level of feature dependency is much harder. The 'old' enterprise integration has created a whole new management science by making processes explicit and visible, so produced high value just from the modelling exercise alone even before starting optimising analyses. So too, value feature extraction and integration has tremendous importance just in getting an enterprise to explicitly and formally examine how the work of the enterprise contributes to the value of the enterprise. The invaluable new asset of fact-based decision making is all gravy after that.

Another comment: trends in the past were to attempt this integration at the quantitative level, reducing everything to numbers and just 'doing the arithmetic'. When one reduces a model to a number or a set of numbers, the result scrubs out most of the intelligence that was put in there. The features described here are models themselves. They are abstracted from the basic models, but retain the important richness, the cause-and-effect dependencies

and functions (some non-linear) that are required for real decision-making, the kind where one can trace the implications in a visible and auditable manner.

6.4.2 Dimension: Agency

Agency in this context is intended to cover two concepts:

- the enfranchisement of a process owner to effectively improve the capability of the enterprise, and
- the ability to speculate about possibilities by simulating them.

Capability improvement only occurs when someone in the enterprise can both *see* a change that needs to be made, and *effect* that change. A low measure of capability improvement results when only a few agents in the enterprise have this power; the opposite is true when many such agents exist. In practice, it is desirable to strive toward each process owner being able to discover changes that would enhance the enterprise (even if they apparently compromise the process) and have the power to make that change happen.

This means that more capable enterprise modelling frameworks will express their models as agents. Simple models are usually passive representations of processes. Complete simple models contain some critical mass of the actual cause and effect physics of the process. Agent models use that cause and effect representation in order to actually simulate the process by executing the model as code[1]. An even more capable model would be used to control the process, effectively replacing much of the cause and effect mechanics.

Agent models have distinct advantages, especially if they can see the value features noted above. Agents can create and evaluate alternative futures by positing a large number of 'what-if' changes. Often these simulations are performed in parallel. Parallel simulations pay off significantly in real enterprises because, most importantly, capability change involves several processes changing in concert. Often, the outcomes are non-deterministic, meaning they are impossible to predict by simple extrapolation.

Fortunately, a great deal of attention has gone into agent systems generally, much of which is directly applicable to enterprise integration for capability improvement and the challenge of collaboration around value features like those outlined above.

A challenge remains: how to manage mid-level selfishness. The problem is that in both supply chains and virtual enterprises, one finds three types of agency levels: processes, component businesses, and the enterprise. What has been described so far is a mechanism for processes to sublimate their selfishness to the whole enterprise. However, this can only be supported if the middle level (the allied business partners) benefit in a strictly selfish way. This is a tricky problem, which can be accommodated at the value feature level by

[1] the complete simple model involved must contain *executable* code understood by the agent models

assigning dollar rewards to partners based on increased value. At the agent level, it can be accommodated by the mechanism of evolved, selfish aggregates of processes. But a general solution to the combined case is only just emerging and at the time of this writing not commercialised.

6.4.3 Dimension: Introspection

Early enterprise integration strategies divided the world into two parts: the world of action and the modelling world. The manager or analyst existed in the modelling world and observed the world of action. But the two worlds are not separate; managers exist in the enterprise. Management infrastructure does too, and in most enterprises the people, resources, knowledge and activities associated with managing work are of the same order as the work itself. Management processes are often in as much, or more need of continual improvement.

In the old 'batch mode' of enterprise integration, the two worlds could be separated. But the notion of continuous improvement includes a concept of enterprise state that necessarily includes *improvement.* Continuous improvement is a slippery, self-referential concept, since it must model, analyse and change the processes of creating, analysing models and triggering / sourcing change (in effect using itself). Information technology infrastructure must be included as enterprise resources, second order resources if they must.

Here again, the introduction of continuous improvement induces new requirements of the modelling methods and frameworks of enterprise integration. Both must be introspective, able to see and reason about themselves. The good news is that *knowledge representation* techniques solved this problem long ago. Old modelling methods must give way to the new more formal ones, but that would be the case anyway with the two requirements noted above.

The trick in most practical systems is for each process to be defined at some standard granularity. This is common in all enterprise integration frameworks. Then these processes are modelled as agents. If that agent 'knows' itself, and if it and all other agents respond to the same value features, then it can know all of the others in the enterprise in a deeply introspective manner.

Readers may recognize many functions and terms from knowledge management that are being used in this evolution of enterprise integration for continuous improvement. Indeed, such systems are subsuming knowledge management functions as a matter of course. And a good thing too, since the primary deficiency of knowledge management systems is the inability to place an element of knowledge in its value context within the enterprise.

6.4.4 Dimension: Federation

This final element is not as necessary for capability improvement as the other three. But it is functionally as necessary because it is demanded by the nature of fluid supply chains and agile virtual enterprises. The notion of federation

is that the models of independent business units are relatively unconstrained. Or rather, they are not under mandates from a central organising authority, though standards are clearly part of the picture. (The process specification language, the basis for federation standards, is described in Chapter 16[2].)
In federated systems, the models are developed in a distributed fashion by the relevant process owners. They remain with these process owners. The primary advantages are two.
A key problem with centralised enterprise models is *keeping them current.* There is a distance between the process owner and the enterprise manager. 'Old' enterprise integration systems were an add-on tax that did not help the process owner. The cost of maintaining a remote useless model meant that there was an inevitable delay in the accuracy of that model. That was an acceptable situation in the batch integration days but untenable in a continually improving situation where everything is dynamic.
The solution is to keep the model close to the work and engineer it so that it helps directly with the work at hand. Extending the model in the ways described above satisfies that requirement, but it also implies that diversity in representations will be common.
The other problem is the matter of this diversity. Integration means that models should be able to be aggregated. Thus, some means must exist to bridge the semantic gap between these models and their framework. Remember that in a continually improving enterprise, the modelling methods themselves continually improve, therefore the source models may not be the same, even from week to week.
Federation is accomplished by leveraging the three techniques noted above (Sections 6.4.1-6.4.3). Each process has agency that knows itself and the system. The mapping to the framework is collaboratively accomplished between the process and the framework as surrogate for the enterprise by ontologies based on value features.

6.5 Conclusion

Enterprise integration originated from a need to understand and engineer processes enterprise-wide. Now, a later generation of enterprise integration methods extend that utility to the support of continuous improvement. The evolution is a continuous one, centred on more robust and capable modelling techniques. These are challenging, but realistic today. Next generation fluid supply chains and agile virtual enterprises are emerging. The Chapters of this handbook provide the tools to support this emergence.

[2] [editors note] A more complete specification of the process specification language can be found in Lee, J., Gruninger, M., Jin, Y., Malone, T., Tate, A., Yost, G. (1998) PIF, The Process Interchange Format. In Bernus, P., Mertins, K. and Schmidt, G. (Eds) Handbook on Architectures of Information Systems, . Berlin : Springer

7

DEVELOPING THE BUSINESS MODEL – A METHODOLOGY FOR VIRTUAL ENTERPRISES

Martin Tølle and Johan Vesterager

Technical University Denmark (Denmark)

7.1 Introduction

Enterprises co-operate with other enterprises in all phases of product life cycle to achieve cost reduction, increase their operational flexibility, and allow them to focus on core competencies. The preferred method for co-operation can be anything from long term alliances between partners in fixed supply chains – to a goal oriented, project focused co-operation as it is usually done in virtual enterprises (VEs).

In the initial phase of the life cycle of an enterprise it is not often clear what is the extent of changes that are necessary, and often it is even less clear what is the list of enterprise entities involved in that change. There are many ways in which an enterprise can conduct business. To carry out a change process successfully, the involved enterprises / enterprise entities need to be identified, and their relations must be understood. These relations will determine the business functions, and they must be chosen carefully to ensure continued operation.

There are two types of relations between enterprise entities in general:

- two enterprise entities co-operate – i.e., in their operational phase they exchange products as well as production- or management related information,
- one enterprise entity's operation covers some life cycle activity(ies) of another enterprise entity (e.g. as part of its operations a company (entity 1) identifies and develops the concept of a new product (entity 2)).

This Chapter presents a methodology to develop a *Business Model* on the example of Virtual Enterprises. This Virtual Enterprise Methodology (VEM) outlines activities to consider when setting up and managing VEs. As a methodology, the VEM helps companies to ask the right questions when preparing for- and setting up an *Enterprise Network*, which works as a breeding ground for setting up VEs. The VEM applies the Virtual Enterprise Ref-

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

erence Architecture (VERA) as an underlying structure. VERA[1] is a specialisation of GERA[2], which is a component of the GERAM [3] Framework (refer Fig. refch3:fig1 and Fig. 2.8 in Chapter 2).

7.2 Introducing the VE Concept

As part of forming an enterprise network, partners need to establish a degree of *preparedness* for forming particular VEs, which in turn will be based on the competencies available in the network in order to meet customer demands. Thus, the network works as a breeding environment for setting up VEs. Additional competencies from sub-suppliers or local contractors may be included in VEs as well (Tølle et al. , 2000) (refer Fig. 7.1). When the customer's requirements are satisfied, the experiences gained in the VE are transferred back to the network, the VE is decommissioned and the network awaits or seeks other possibilities in the market.
While from the customer's point of view a Virtual Enterprise is functionally identical to a company, the VE may be in fact:

- a temporary association of companies to perform either a one-of–a-kind task (for building a one-of-a-kind product (OKP) such as building a bridge, or a ship), or
- an association of companies towards the purpose of performing some sustained task during a period of time (such as producing a large batch of products, or to perform maintenance services of a product for the duration of the product's life).

For this reason, the concept of Virtual Enterprise may be considered as *a generalisation of the concept of 'Project'*. To distinguish between the one-of-a-kind VEs and the sustained production / service delivery VEs, the former may be referred to as a Project VE (or project enterprise), and the latter as a Production VE or a Service VE[4].
While the concept of Project has been around for a long time, it is enlightening to look at a project as an enterprise, because this view gives a complete

[1] Both VEM and VERA were developed as part of GLOBEMEN (Global Engineering and Manufacturing in Enterprise Networks) - an international IMS (Intelligent Manufacturing Systems, http://www.ims.org) project. Refer to the Technical Research Centre of Finland (VTT)'s Globemen partner web site (http://globemen.vtt.fi/) for details.

[2] acronym for Generalised Enterprise Reference Architecture

[3] Generalised Enterprise Reference Architecture and Methodology, (ISO/TC184/SC5/WG1 , 2000)

[4] this terminology is consistent with the GERAM classification of Operation-oriented Enterprise Entity Types (refer (IFAC/IFIP Task Force , 1999) and Chapter 2, Section **3.1.3.3.1** for details)

life-cycle view of projects and their relations to other enterprise entities. Similarly, while the manufacture of a product involves an (ideally, *integrated*) supply chain, creating the supply chain as one (virtual) enterprise allows the life-cycle view to be employed in designing a well managed, integrated supply chain. This understanding of VEs allows companies to develop improved competencies in project design and management, as well as supply chain design and management. Furthermore, these competencies are generic (reusable in many business design and management situations faced by a company) – thus their development pays off. VE design and management competencies must be developed as core competencies themselves, because of their obvious relation to the production or service delivery core competencies of a company (as an enabler to use these value adding competencies).

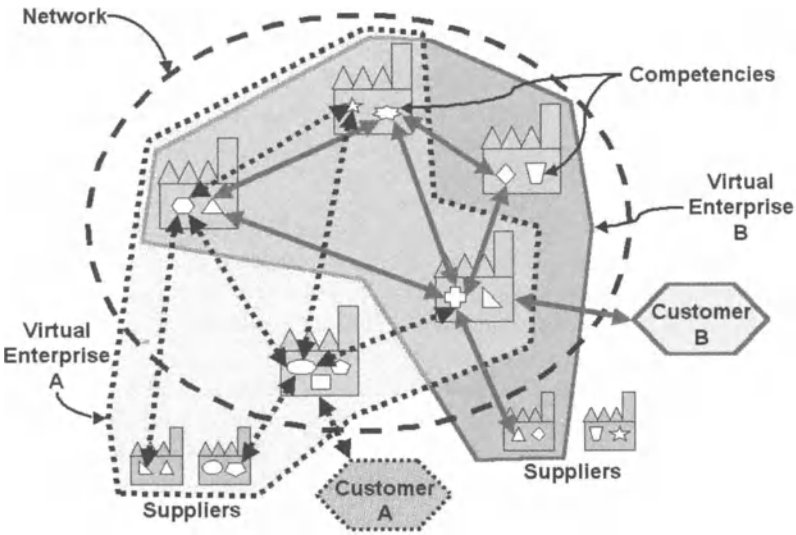


Fig. 7.1. A rich picture of the VE concept (Globemen Consortium)

7.2.1 Preparedness – Reference Models and Methodology

A central and important concept for setting up VEs in a continuous, fast and dynamic way is called '*preparedness*'. The potential competitive advantage of a VE – being able to configure world class competencies together into a system of service delivery or production – is often jeopardised by the time it takes to setup a VE, especially if the VE is composed of partners unknown to one another before the formation of the VE (Tølle et al. , 2002a).

When setting up an enterprise network, one of the key questions to consider is the level and type of preparation in the network and among the network

partners. *Reference Models* (RMs) are important means for this preparation. A RM is a model that captures characteristics and concepts common to several entities, e.g. networks or VEs (Tølle et al. , 2002a). The purpose of RMs is to "capitalise on previous knowledge by allowing model libraries to be developed and reused in a 'plug-and-play' manner rather than developing the models from scratch" (Chapter 2 and (IFAC/IFIP Task Force , 1999)). Hence, RMs are deemed to "make the modelling process more efficient" [ibid.]. RMs are used to convert the VE formation task into a 're-use/configuration' task resulting in quick, low-cost, and secure VE creation.
Many different RMs exists today. Tølle et al. (2002a) have mapped eight different RMs applicable for VEs. What is lacking however, is a *methodology* helping the selection of applicable models and describing how to handle the management task of creating an enterprise network capable of being managed effectively and efficiently (i.e. within a reasonable timeframe and capable of delivering a competitive solution to the customer). This Chapter aims at providing a methodology for the latter.

7.3 VERAM

As a part of the IMS Globemen project, a Virtual Enterprise Reference Architecture and Methodology (VERAM) has been created, based upon GERAM. The purpose of VERAM is to structure a body of knowledge that supports future work in the area of global engineering and manufacturing of enterprise networks.
A part of this knowledge is in fact similar and common every time a VE is setup or operated, and could be standardised and re-used (Zwegers et al. , 2001, 2002). VERAM positions elements that support modelling, formation/setup, management and ICT support of VEs, such as reference models, and supporting tools and infrastructures. The relations among these elements are indicated in Fig. 7.2.

7.3.1 VERA

Similar to GERAM, one of the central parts of VERAM is a reference architecture called the Virtual Enterprise Reference Architecture (VERA, in Vesterager et al. (2002)). Figure 7.3 shows how VERA captures the VE concept described above. VERA consists of three recursive entities: a network, VEs and product(s). Once management decides that it will explore the possibility of using VEs to provide services, or to produce goods, the *types* of enterprise entities involved is determined, therefore in VERA one no longer refers to enterprise entities in general, but to enterprise network(s), partners, suppliers, virtual enterprises and products (each of the above being an enterprise entity). The figure shows that the network in its operational phase

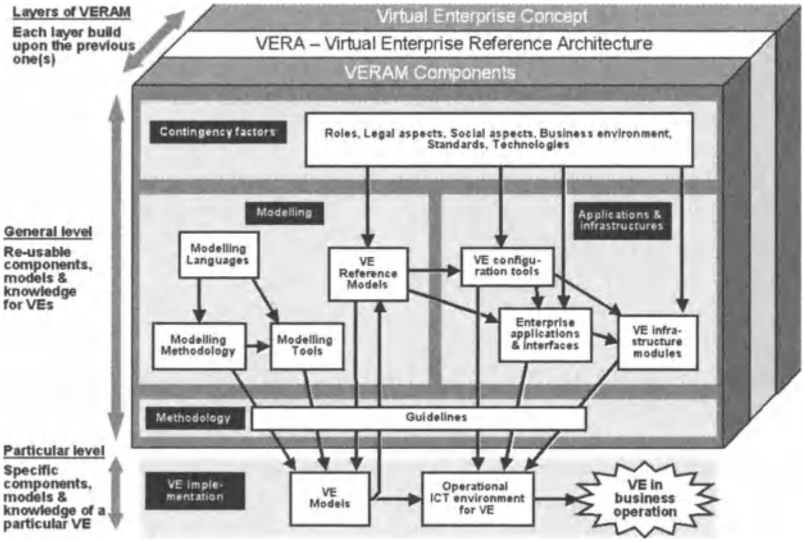


Fig. 7.2. VERAM Framework – Virtual Enterprise Reference Architecture and Methodology (note the similarity between this figure and Fig. 2.1 in Chapter 2)

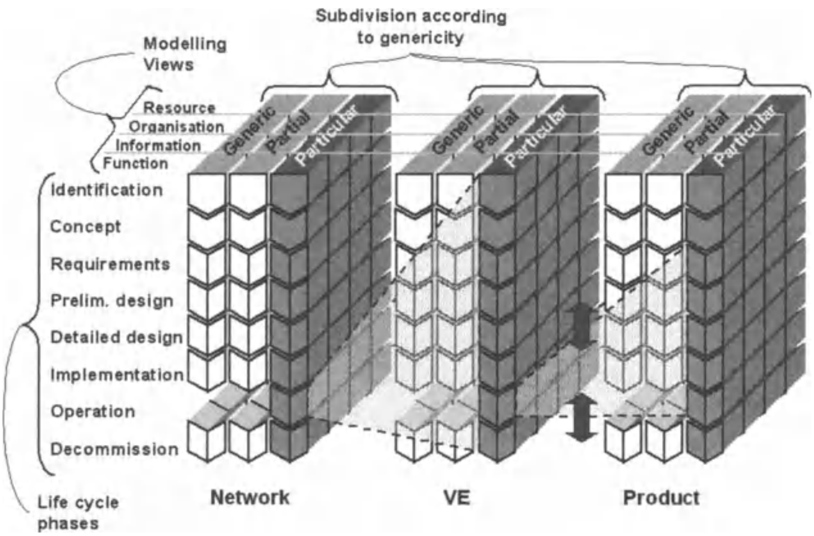


Fig. 7.3. VERA – Virtual Enterprise Reference Architecture (based on Vesterager et al. (2001a)

creates VEs and a VE carries out some product life cycle activities ('phases'), indicated by double arrows.
Each of the three entities is represented by use of the three dimensions of the GERA modelling framework: the life cycle dimension, the genericity dimension, and the view dimension (Chapter 2). The rightmost, dark part of VERA in Fig. 7.3 represents the *particular* part, i.e. the part that is related to the specific network, VE and product respectively, whereas the white boxes represent the generic/partial (and thus reusable) parts. Thus, RMs – such as the ones mapped in (Tølle et al. , 2002a) – are located in the *white* part of VERA and, when used, they are instantiated into a particular case (e.g. when setting up a VE).
Another view applied in VERA is the explicit inclusion of the so-called Purpose View of GERAM, which distinguishes between the *customer product and service activities* and the *management and control activities* representing the mission fulfilment and the mission control aspects respectively (refer Chapter 2 and (IFAC/IFIP Task Force , 1999)). This distinction is made starting from the requirements phase to the decommission phase of an entity (as shown in Fig. 7.3). More elaborate descriptions of VERA can be found in (Tølle et al. , 2000) and (Vesterager et al. , 2001a,b). In this Chapter, VERA will be used as an underlying structure for the VEM.

7.4 Life History Example

Before describing the VEM, the *life history*[5] of an enterprise network is explained, i.e. how the network and its VEs evolve over time. It should be noted that this is only one of many possible life histories. The example integrates the activities of the three VERA entities together into one life history description [6]. Ollus et al. (2002) describes a so-called demonstration scenario outlining different versions of an enterprise network and illustrate how the prototypes developed by the industrial partners can support different activities related to setting up the network and its subsequent setup of a Quotation VE, Project VE and Service VE.
In brief, this life history example starts with the identification of a network. This is followed by the preparation and engineering of the network. At some point in time, during the operation phase of the network, a customer contacts the network with a request for quotation. Based on the customer's need, the network sets up a Quotation VE that carries out the first phases of the product's life cycle, in order to deliver a quotation to the customer. The customer accepts the quotation and the network consequently sets up a Project VE,

[5] life history represents the succession of life cycle phases that a (virtual or not) enterprise has-, and/or is reasonably expected to go through during its existence ((IFAC/IFIP Task Force , 1999; ISO/TC184/SC5/WG1 , 2000)

[6] Other life history examples can be found in (Vesterager et al. , 2002; Ollus et al. , 2002)

which completes the product design and finally produces (builds) the product. Figure 7.4 illustrates the overall picture of the life history of the network. It should be noted that, compared to Fig. 7.3, the three entity life cycles are arranged *above* each other, and also that a *time* dimension has been added (time elapsing from left to right).

The triangles in Fig. 7.4 symbolise RMs such as models, tools and procedures created by the network or VEs, which support activities in the operation phase of either the network (e.g. contract models) or VEs (e.g. product models). Numbers (1 to 16) refer to the description below:

1. At time $\mathbf{T}_0$ an *initiator* related to the *initiating company* identifies that the company could benefit from establishing a more formalised enterprise network.

2. After a preliminary feasibility study, the *initiator* presents the idea of a more formalised enterprise network to his/her superior. This 'internal recognition' process includes convincing the CEO of the company and getting CEO and management commitment to carry on with the project. In the process of reaching this recognition, one or more internal meetings and workshops may be held, resulting in a formulation of the network concept. Following the 'internal recognition', the *initiating company* contacts potential network partners and informs them about the network idea. Furthermore, the *initiating company* prepares one or more network kick-off meetings.

3. At the kick-off meeting(s) the opportunities of establishing a more formalised enterprise network are discussed. The partners agree on the concept of the network (including the description of overall purpose, types of VEs, types of products, and markets to address). Furthermore, a list of preparation projects is specified. This includes the definition of overall requirements for each project, such as clarification of overall business rules of the network, overall contractual / IPR[7] issues, the type of tools, procedures, etc. to be used. At the end of the kick-off meeting, a set of preparation project groups has been defined.

3a. Additionally, the partners establish a *network preparation management team*, which manages the further preparation and setting up of the network, i.e. subsequent activity #3a. This *network preparation management team* is in operation at time $\mathbf{T}_1$. In Fig. 7.4 this is shown as the management and control system of the network.

4. Each of the groups carries out their preparation projects, i.e. carry through the requirements to implementation activities of their specific network projects. These could be projects concerning IT infrastructure, tools (e.g. product configurators), business process models (e.g. order fulfilment process models), legal/contractual issues, organisational issues, knowledge management, quality management, environmental management, standardisation issues, and so on. Figure 7.4 illustrates that some of the preparation projects primarily deal with either customer service and product activities or manage-

[7] acronym for Intellectual Property Rights

ment and control activities, and others deal with both purpose views. The preparation projects create RMs (models, tools, procedures, contract models, etc.) to be applied in the operation phase of either the network or VEs, cf. the triangles on Fig. 7.4.

5. As soon as the different project groups in #3a have concluded their work, the network goes into operation at time $\mathbf{T}_2$ and awaits customer requests. At that time, the focus of the management group defined at #3 shifts from managing the preparation of the network to managing its actual operation.

6. Independent of the operation of the network, a customer identifies a need for a product.

7. At time $\mathbf{T}_3$ the customer contacts the network and initial negotiations are made between the network and the customer. The network sets up a Quotation VE. This includes selecting partners (network partners and maybe temporary, non-network partners) and defining a management structure (selection of *Contract Manager*, contribution of each partner, etc.). At time $\mathbf{T}_4$ the management team of the VE is in place and takes over the further setup (#8) and operation (#9 - #11) of the Quotation VE.

8. The management groups of the Quotation VE can setup one or more project groups each focusing on different aspects of the preparation and creation of the VE. This corresponds to the above-mentioned types of project groups in #4, although the VE projects will typically be more VE specific. Again, topics for project groups could be IT infrastructure, tools, business processes, legal/contractual issues, organisation, etc. It should be noted that the type and extent of preparation projects related to the setup of a VE can vary widely depending on the preparation level already achieved during the network preparation and setup. For a network with a very high preparation level the VE setup might only involve the instantiation of different, already predefined, models – *e.g.* contract models, risk sharing models, business process models, etc., perhaps with some tailoring of these to the particular case. At this stage in time human resources and organisational units, subcontractors are allocated to tasks, as well as software and hardware modules are installed according to the above models.

9. The VE goes into operation at the time the project groups in #8 have concluded their work. In the operation phase (at time $\mathbf{T}_5$) of the Quotation VE the VE-partners transform the customer's need into a product concept, clarify important requirements and corresponding relevant design aspects.

10. This forms the basis for a completed quotation at time $\mathbf{T}_6$. It should be noted that this is only an example; the contents of a quotation could differ from industry to industry.

11. The Quotation VE has completed its tasks and is decommissioned.

11a. As part of this activity the experience with the Quotation VE is collected. This triggers a need for modifying some of the existing RMs and creating new ones.

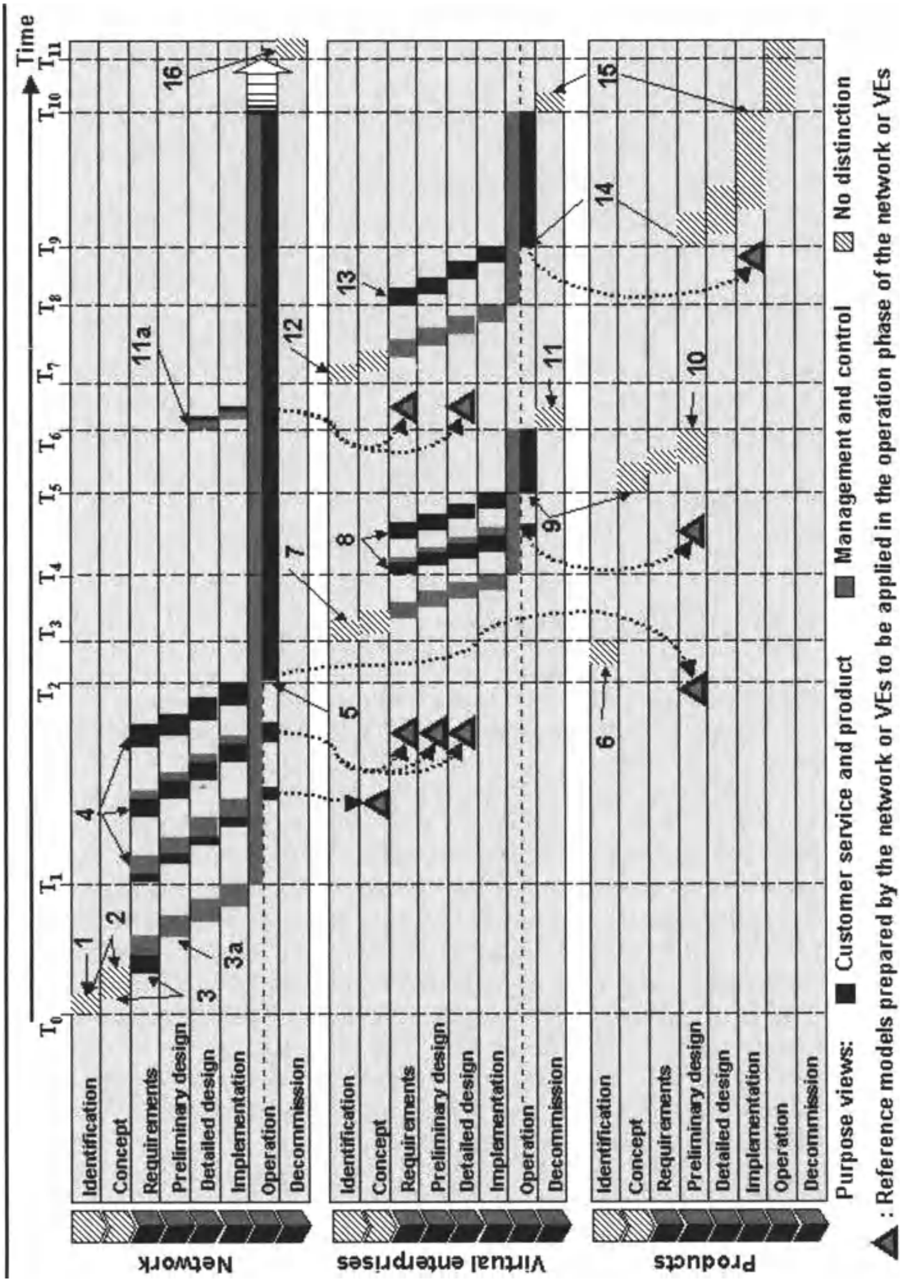


Fig. 7.4. Life history of a network setting up VEs for quotation and production

12. The customer accepts the quotation at time $\mathbf{T}_7$, and the network sets up a management group for a Project VE, which is in operation at time $\mathbf{T}_8$. This corresponds to setting up the Quotation VE (#6).
13. Corresponding to the groups mentioned in #8, the VE management can setup one or more project groups focusing on particular Project VE aspects, e.g. specification and installation of new production equipment. Thus the Project VE may have to create more (sub)project enterprises (this is not shown on the figure).
14. At time $\mathbf{T}_9$ the Project VE goes into operation and finalises the design of the product as well as manufactures (i.e., implements) the product.
15. The product is delivered to the customer for operation at time $\mathbf{T}_{10}$ and the Project VE is decommissioned, which again could lead to the modification of existing RMs (not shown in the figure). At this time a Service VE (not shown in the figure) could be established for service of the product in its operation phase. Similarly, a Decommission VE (not shown on the figure) could be established when the product is to be decommissioned.
16. At time $\mathbf{T}_{11}$ the network is finally decommissioned.

7.5 VE Methodology (VEM)

The life history in the previous section is – as already mentioned – only one example out of several possible ways in which activities can unfold over time. This section will describe key activities in a VE methodology (VEM). Compared to the life history, the description of the following activities contains no reference to time. Thus, the user of the methodology has to determine which activities are relevant to their specific situation. Furthermore, the user needs to add a timeline to the activities, determining their execution sequence. Some guidance may be found in the life history presented in the previous section, however keeping in mind that the activities can unfold in many other ways.
To plan such a complex process, a useful technique is to first create an activity model of network- and VE creation, so as to help design the correct sequencing and information flow of the process. The plan of this process can then be represented on the timeline, where the production of major deliverables (identified in the activity model) is 'pegged out' with milestones and decision points (Bernus, Noran and Riedlinger , 2002).
The first part of the methodology is addressing the life cycle of the enterprise network outlining activities to be considered when setting up and managing enterprise networks. The second part is correspondingly addressing the setup and management of VEs.
It should be noted that the guidelines are relevant for *any* enterprise (and not only for e.g. enterprises not yet operating with partners in a kind of enterprise network). Enterprises already operating with partners (e.g. in a strategic alliance) can still benefit from going through the outlined activities. The starting point of the enterprise, i.e. the 'as is' situation from which the

application of the methodology starts, can be one of many. Two overall types of use of the outlined guideline can be envisioned:

General use (Step-by-step): Using the list stepwise, from top to bottom can support an enterprise in mastering the setup of a more formalised enterprise network. This could also be an enterprise already operating in some type of network environment, but where a need for further preparation and clarification is needed in order to be able to setup effective VEs efficiently, i.e. in a competitive manner with regards to time, cost and quality. Alternatively, it could be an enterprise that is exploring its competencies in a new way, by applying existing competencies in new types of solutions ('business models') where new and unknown competencies and thereby new partners are needed.

Selected use (pick relevant activities): As an alternative to using the whole list step-by-step, selected activities can also be beneficially used. This could be the case if a more formal enterprise network has already been established. In this instance, the list in general may be used as a checklist and serve as an inspiration for how the network could be reconfigured or prepared in a different manner. If a need for reconfiguration is identified, then the list could be used in more detail to support one or more of the described activities.

7.5.1 Setup and Management of Enterprise Networks

Figure 7.5 depicts key activities related to setting up and managing an enterprise network. In the following, each of the activities will be described (refer Table 7.1). It should be noted that in addition to this description, (Tølle et al. , 2000) contains further detail regarding the identification, concept and requirements phases of enterprise networks.

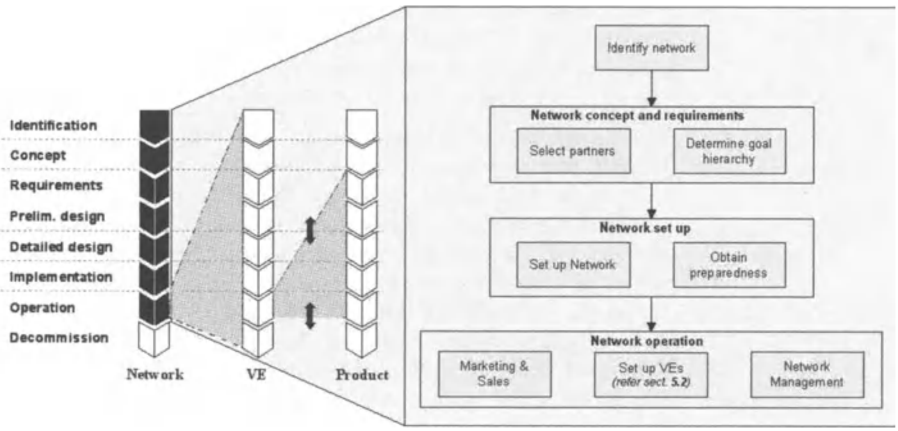


Fig. 7.5. Key network activities (derived from Tølle and Vesterager (2002b))

Table 7.1. Enterprise network setup and management activities

Activity	**Description**
Identify network	The aim of the network identification phase is to identify the overall purpose of the network including its *raison d'être*, the network type, and its boundaries in relation to the internal as well as the external environment. The main outcomes are to identify key drivers, motivations for the enterprise network and to get a clarification of the overall purpose of the network including which type(s) of market(s) it shall address and with which type of products.
Select partners	A central element when preparing for and setting up an enterprise network is to identify and select the partners that should participate in fulfilling the vision of the enterprise network. The selection of partners depends not only on the expected future situation, but also on the existing partnerships of the enterprise driving the network setup, i.e. partners already trusted. To participate in a VE, companies should basically possess two types of competencies: *Functional competence* and *Alliance competence* (refer Pedersen and Berg (2001)). That is, partners should not only posses sufficient competencies to carry out the necessary production and service tasks, but also they should posses the ability to enter into and participate in VEs, e.g. manifested by an ability to manage and implement alliances and ability to display alliance spirit and behaviour (Pedersen and Berg , 2001).
Determine goal hierarchy	To avoid potential conflict among network partners, efforts should be made to establish and ensure that the partners have a shared goal hierarchy, i.e. their mission, vision, strategies and objectives are *aligned.* If it turns out that network partners are pursuing different (or, in the worst case, conflicting) goals, then the success of the enterprise network could be jeopardised.
Setup network	A key challenge when operating in an enterprise network is to be able to setup competitive VEs within a short timeframe. One of the key means for doing this is to prepare the network partners in order to enable the configuration of customer-focused VEs faster and more efficiently. Many different elements of a network can be prepared (see examples in #4 in the above life history), and the type and level of preparation for a specific enterprise network depends of the type and frequency of tasks that the network expects to carry out. Once the aimed 'to-be' situation is determined, suitable actions has to be taken in order to evolve each of the partner enterprises from their existing 'as-is'- to their desired ('to-be') state.

Activity	Description
Obtain preparedness	When the network has determined what type and level of preparedness it wants to pursue and the models have been identified and/or prepared, the decisions / models have to be implemented in the network *and* in every partner, respectively. This may include e.g. installing any needed information and communication technology (ICT) systems and integrating them with the partners' legacy systems, as well as training of personnel, etc.
Marketing & sales	During the operation of the enterprise network, marketing and sales activities have to be carried out. This includes e.g. seeking new customers; responding to customer requests; contract negotiations with customers; marketing activities (i.e. proactively seeking customers).
Set of VEs	The most important task of a network however, is to setup VEs answering customer needs. This is elaborated more in the next section (refer Fig 7.5 and Table 7.2).
Network management	Management of enterprise networks includes all types of management tasks and levels known from traditional management of conventional enterprises. This includes e.g. *direct* and *indirect* monitoring, and operational, tactical and strategic level decisions. • Direct monitoring includes operational level progress monitoring and taking appropriate actions to manage the goals pursued and achieved, e.g. setting up VEs able to respond effectively to a customer need, or solving possible disputes among partners. • Indirect monitoring looks at the appropriateness of e.g. the network's level and type of preparedness and initiates appropriate actions if a need for reconfiguration is needed. Reconfiguration of networks/VEs is elaborated further in (Vesterager et al. , 2001b, 2002)

7.5.2 Engineering, Operation and Decommissioning of VEs

As previously mentioned, the most important activity and *raison d'être* of the network is to be able to timely deploy and operate VEs. Figure 7.6 describes the activities involved in the operation of networks, i.e. the engineering, operation and (if applicable) decommissioning of VEs. Each of the activities in Fig. 7.6 are described at a high level in Table 7.2.

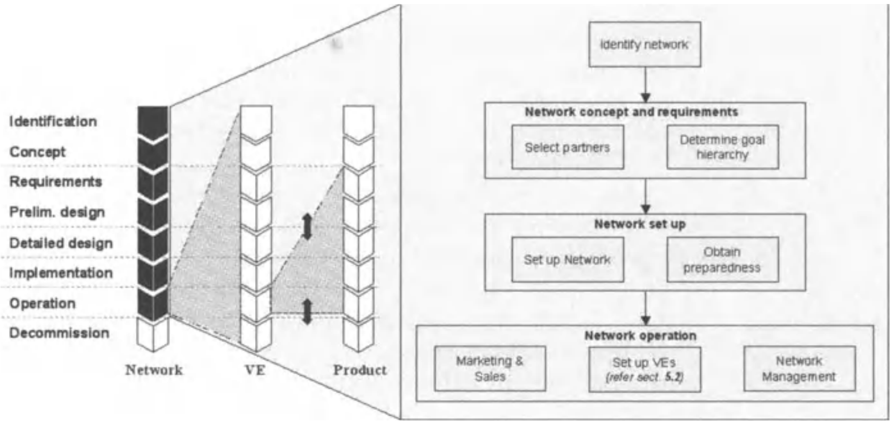


Fig. 7.6. Key VE activities (derived from Tølle and Vesterager (2002b))

Table 7.2. Virtual enterprise setup and management activities

Activity	Description
Analyse customer requirements	When a network partner is addressed with a customer need, the first activity is to assess if the network should pursue the fulfilment of the request, i.e. to clarify if the request is within the scope of the enterprise network.
Select partners	Selection of partners to participate in specific VEs includes assessing if partners possess s ufficient capabilities (i.e. have knowledge and experience to fulfil the type of work) and also if they have the necessary capacity *at the requested time*. Most of the considerations about partner selection in the network are also valid when selecting VE partners, as shown in Table 7.1. Of course, the more extensively the partners have been assessed in the network, the less assessment needs to be done as a part of the VE, and vice versa.

Activity	Description
High level WBS	Parallel with the selection of partners for the VE a Work Breakdown Structure (WBS) should be created. The WBS captures the decomposition of the VE's product into deliverables and the accompanying partner selection, i.e. which partner is responsible for which deliverable.
Setup VE	The setup of a VE highly depends on the type and level of preparation work already determined as a part of the network setup (Table 7.1). If the network has been foreseen and is prepared for the configuration of various VEs, then less has to be decided / created during the VE setup[8] The setup includes: • Set up VE infrastructure, e.g. a multi tier (partner) project structure, define access rights and interfaces with partners' legacy systems • Determine VE rules, templates to be used, RMs to instantiate in the specific situation and other tools. • Resolution of contractual issues, e.g. use of contract reference models and their tailoring. • Creation of the VE organisation, e.g. assignment of human resources, organisational groups, contractors and subcontractors to standard roles. This may be done in a predefined way (based on pre-qualification of contractors and suppliers), or in an ad hoc manner (without the reliance on pre-qualified contractors and suppliers), depending on the nature of the specific project enterprise.
Obtain preparedness	This activity is similar to the corresponding activity of the network (refer Table 7.1). Obviously, the more done during the network set up, the less is left to do during the setup of the VE.
Planning & scheduling	Once the VE starts operating, more detailed planning is needed. Each of the partners have to make more detailed plans of what they will do, schedule the VE's tasks in accordance with their other activities and decide if they will sub-contract some of the parts to their own suppliers if needed and not already decided. Further details of this can be found in (Tølle and Vesterager , 2002b).
VE management	Similarly to the network management the management of a VE includes *direct* as well as *indirect monitoring* (refer Table 7.1).

[8] Ideally, this task should in fact become an *instantiation* of previously prepared models.

Activity	Description
Monitoring & assessment	Monitoring of projects is, together with progress reporting, an important project management activity, ensuring that the project is completed within time and budget. VEs partners should continuously assess the genericity (in the sense of reusability) of the tasks performed and assess if the current type and level of work preparation in the VE and enterprise network is sufficient, or needs to be modified or updated. This includes: • Assess if the RMs / technologies / standards / procedures / rules of the network and VEs are sufficient or if new ones are emerging / needed? • Assess if a task performed within a VE is of a certain genericity that could make it appropriate to adopt it at the network level for future similar projects.
Experience collection	Once the product has been handed over to the customer and all paperwork and payments have been taken care of, it is time to decommission the VE. However, before closing it down, the partners should consider capturing their experiences learned in the project. Activities may include: • Fill in 'log-book' available for VE partners; • Fill in 'log-book' available for all network partners (probably a subset of the VE log-book items). • Coverage of topics such as: – What have we learned; – How would we do it different in the future; – Is there a need for updating or changing the type and level of preparations in the enterprise network?
Close down	The project is closed down. The partners go back to the enterprise network and await new customer needs.

7.6 Conclusion

The Virtual Enterprise Reference Architecture (VERA) presented in this Chapter provides a generic structure, which permits a systematic approach to the complex and multi-dimensional tasks involved in creating Virtual Enterprises (VEs). To support the realisation of VEs there is a need for a comprehensive methodology for VE engineering and management.
The presented Virtual Enterprise Methodology (VEM) applies VERA as an underlying structure. The methodology focuses on setting up and managing enterprise networks and virtual enterprises (VEs). The methodology is intended to support enterprises that are faced with the challenge of operating in accordance with the VE concept presented in this chapter, i.e. in a more

dynamic and agile matter. The VEM can help these enterprises to 'ask the right questions at the right time', and as such facilitates the planning and preparation of VEs.
Applying a GERAM-based architecture as VERA as an underlying structure for the VEM reduces the risks of misunderstandings and permits dissemination to a broader audience. Furthermore, a shared reference architecture as VERA makes it possible to focus on a subset of the VE challenge while still securing integration of work carried out by different partners. Thus, VERA allows the unification of research and practice within the VE area.
VEM addresses activities of relevance when setting up and managing enterprise networks and VEs. VEM integrates existing methods and procedures into a VE context, i.e. most of the management activities are well known methods and procedures – what is new is that they are put into the VE context. The VEM presented in this Chapter can work as an important means to widespread realisation of more agile virtual enterprise type of organisations.

References

Bernus, P., Noran,O., Riedlinger,J. (2002) Using the Globemen Reference Model for Virtual Enterprise Design in After Sales Service. In 'Global Engineering and Manufacturing in Enterprise Networks, GLOBEMEN, VTT Symposium 224' ed. I Karvonen, R van den Berg, P Bernus, Y Fukuda, M Hannus, I Hartel, J Vesterager, VTT Information Service, 2003, pp. 91-110.

IFAC/IFIP Task Force on Architectures for Enterprise Integration (1999) GERAM V1.6.3: Generalised Enterprise Reference Architecture and Methodology, ISO15704:2000. (available on: http://www.cit.gu.edu.au/~bernus)

ISO/TC184/SC5/WG1. (2000). Annex A: GERAM. In ISO/TC184/SC5/WG1, ISO/DIS 15704: Industrial automation systems - Requirements for enterprise-reference architectures and methodologies.

Ollus, M., Hartel, I., Tølle, M., Wubben, H., Hannus, M. (2002) Using scenarios for dissemination, Experiences from the IMS GLOBEMEN project, Proceedings of eBeW2002 conference, Prague, 16-18 October 2002

Pedersen, J.D., Berg, R.J. van den (2001) Supporting partner selection for Virtual Enterprises. In J. Mo, L. Nemes (Eds.) 'Global Engineering, Manufacturing and Enterprise Networks' , Kluwer, 2001, pp.95-102

Tølle, M., Vesterager, J., & Pedersen, J. D. (2000), A Methodology for Virtual Enterprise Management – Results from IMS 95001/Esprit 26509 Globeman21 project, Proceedings of the 6th International Conference on Concurrent Enterprising, Toulouse, 28-30 June 2000, pp. 119-127.

Tølle, M., Bernus, P., Vesterager, J. (2002a) Reference Models for Virtual Enterprises. In Luis.M. Camarinha-Matos (Ed.) 'Collaborative business ecosystems and Virtual Enterprises' , Kluwer 2002, pp.3-10

Tølle, M., & Vesterager, J. (2002b) Guidelines for Inter-Enterprise Management (IEM), Globemen Deliverable D23, 2002 (http://globemen.vtt.fi)

Vesterager, J., Bernus P., Larsen L.B., Pedersen J.D. & Tølle M. (2001a) Use of GERAM as Basis for a Virtual Enterprise Framework Model, in J. Mo and L. Nemes (Eds.) 'Global Engineering, Manufacturing and Enterprise Networks' , Kluwer, 2001, pp.75-82

Vesterager, J., Bernus, P., Pedersen, J.D., & Tølle, M. (2001b) The what and why of a Virtual Enterprise Reference Architecture, In B. Stanford-Smith, E. Chiozza (Eds.) 'E-work and E-commerce, Novel solutions and practices for a global networked economy' IOS Press, 2001, pp.846-852

Vesterager, J., Tølle, M., Bernus, P. (2002) VERA: Virtual Enterprise Reference Architecture. In 'Global Engineering and Manufacturing in Enterprise Networks, GLOBEMEN, VTT Symposium 224' ed. I Karvonen, R van den Berg, P Bernus, Y Fukuda, M Hannus, I Hartel, J Vesterager, VTT Information Service, 2003, pp. 39-52

Zwegers, A., Hannus, M., Tølle, M. (2001) Integration issues in virtual enterprises supported by an architectural framework, In Bernardo Ferroni, Claudio R. Boer, Charles M. Anderson (Eds.) 'International IMS Project Forum 2001, A common platform for cross-fertilization among IMS projects and promotion of the IMS program' , ISBN 88-900298-3-8, 2001

Zwegers, A., Tølle, M., Vesterager, J. (2002) VERAM: Virtual Enterprise Reference Architecture and Methodology.In 'Global Engineering and Manufacturing in Enterprise Networks, GLOBEMEN, VTT Symposium 224' ed. I Karvonen, R van den Berg, P Bernus, Y Fukuda, M Hannus, I Hartel, J Vesterager, VTT Information Service, 2003, pp. 17-38

8

ANALYSING THE PRESENT SITUATION AND REFINING STRATEGY

Greg Uppington[1] and Peter Bernus[2]

[1] Exeige (Australia)
[2] Griffith University (Australia)

8.1 Introduction

The audience for this Chapter ranges from the CEO to the technical specialist (in-house or consultant), i.e. stakeholders who will be major players in the change methodology and are identified by their role in the change process.
This Chapter is placing much emphasis on various types of preliminary analysis deemed necessary to support enterprise change processes. Many of these processes are strategic in nature and thus there is a need for 'strategic input' on behalf of management with leadership abilities (refer to Chapter 5). Often, the various strategic alternatives need to be analysed to establish which one is realistically attainable organisationally, technically and financially. The analytic processes described herein are *not a replacement* for strategy making but are indispensable for *supporting* strategy making, so as to ensure that the developed strategy is feasible.
In this chapter we first expose the need for identification activities, and then provide a technique by which an individual enterprise may determine which of these tasks are actually necessary for the particular enterprise, given a change process in question. Some notable techniques not addressed elsewhere in this book are presented in more detail (such as benchmarking and maturity assessment), while other techniques that are similar to techniques applied in later life-cycle phases are only referred to. It is to be noted, however, that models developed in support of strategy confirmation are much less detailed than design models because the main reason for modelling is the identification of obstacles, problems or opportunities and not the creation of a new design.

8.2 The Importance of a Clear Direction: The Identification Phase

The one thing that all current and future enterprises have in common is a *current situation.* Even future or green-field enterprises are still modelled or based

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

on certain existing structures, whether they be organisational, IT, cultural, etc. Within that current situation are people, processes, mission, management, organisational identity and a reason for existence. It is knowledge of the current situation and its comparison to other internal and external elements that is the catalyst of any desire for improvement. The process of creating a sufficiently detailed understanding of the current situation is usually called *as-is analysis.*

There are many arguments for and against conducting a detailed as-is analysis (e.g. Goodstein et. al., in (Pfeiffer , 1991), but there are few arguments that propose to have no as-is analysis whatsoever. One of the purposes of the identification phase is to understand the business functions, the organisational beliefs, values, cultural and political factors, as well as problem areas and then contrasting the actual organisational mission against these findings. The as-is analysis allows and provides a basis for understanding, consensus and planning as well as insight into the unwritten but underlying business assumptions.

The need to understand what is occurring in an enterprise before changing it requires those in positions of influence and decision to understand the current business practices before they start questioning the effectiveness of such practices. "Most of the primary threats to survival and vitality in organisations develop slowly, and they are not caused externally" (Kofman and Senge , 1993, p.11). An as-is analysis can provide the knowledge and understanding of critical activities and cultural values that must be incorporated into the desired future scenario.

If the as-is analysis is constructed in a participative fashion, it will have many sets of fingerprints on it to help secure psychological buy-in to the effort as a whole (e.g. (Jick , 1995)). The Purdue Handbook on Master Planning expresses the view that the initial internal analysis group should be as small as possible, using a small number of employees from a cross section of functions and levels to represent the organisation, so as to avoid any detrimental effects of rumours, etc. before the need for change is established (Williams et al. , 1996). Others advocate using feedback from corporate and stakeholder meetings or engaging all staff in the process. Regardless of the makeup, cross-functional and multi-levelled team members (including outsiders) must be brought to a common level of understanding about the process in order to contribute equally to the development of the assessment. This will also give inertia for conducive team commitment, management buy-in, and decrease cultural resistance (e.g. Spector (1989)).

If the purpose is radical change, such as described in Hammer and Champy (1993), a recommendation of re-engineering is not to analyse the present situation. However, the question arises: without a baseline understanding, how can a change be justified?

The customer is the most important external aspect of any organisation. Without an adequate understanding obtainable through an as-is analysis, problems will arise in the future to-be scenario due to inadequate capture of internal or external customer needs (Kofman and Senge , 1993). One should there-

fore explicitly know the interactions between the customer and the enterprise ('Enterprise Business Entity', or EBE). From such an analysis, it will then be possible to understand the process a customer goes through in order to do business with the company, and the existing processes which provide the enterprise's products or services to the customer.

As different industry sectors have different levels of labour intensity, the level of understanding of the current practices depends much on the industry sector in which the company operates. In labour intensive industry sectors (such as the hospitality industry) the human organisational component is mostly unknown in any formal way that is easy to analyse (for example, in the documentation of external customer contact). The level of labour-intensity within an organisation therefore seems to be proportional to the level of analysis usually required to understand what is really happening in the organisation. When interfaces are not defined, or work-related cooperation among personnel is very informal, little may be known about the actual business processes, because many processes are internalised and exercised (and possibly- but not necessarily explicitly known to those who carry out these processes). In highly developed and automated process industries, the current situation and as-is models are mostly well known, because without these models no automatic control would be possible in the first place. In the more labour-intensive discrete-part manufacturing however, the current as-is models are known to a lesser extent. In general, in highly labour intensive or high innovation content activities the processes may be far less obvious even to those who exercise them.

The PERA reference architecture and the GERAM life-cycle model (refer to Chapters 2 and 3) call the top enterprise life-cycle activity the 'identification phase', in which an Enterprise Business Entity is appropriately identified as the target for change[3] It is essential to identify the organisational scope and boundaries of an EBE analysis and self-assessment for setting and aligning organisational directions and resource use at the work unit, key process, departmental, and whole organisation levels.

---

[3] It is not predetermined what should be the size and extent of an EBE; it can be the entire organisation, or any part of it that management want to consider as a co-operating but autonomous unit. Typically, an EBE may be a whole business unit responsible for a type of product, product group or service. This business unit may be situated at a given location, or distributed within a region or globally. The best way to delineate the target EBE is by considering the service the EBE gives to the market and to its environment. All activities which are considered by management to be core (strategic) activities in the provision of the above service, are ideally handled within the EBE. Other, non-core activities are either done in separate EBEs of the same enterprise or are performed by other enterprises. In this way, an enterprise of any size can be considered an EBE, comprehensively managing all activities that are needed in the value chain of its products, and which activities are carried out in inter-enterprise or intra-enterprise cooperation (i.e. between EBEs)

8.3 Significant Events and Change Initiatives

A change initiative may originate from any enterprise area at any stage of an enterprise's life history. A change initiative originates from the operation enterprise entity and is usually due to some *'significant events'*. The origin of the event may be on any level of management or the operations, and can be either external or internal to the organisation.
What makes an event significant is not its scale, but the known or potential consequences of it on the operation of the enterprise. E.g. while the breakdown of important machinery in a supplier's factory may cause problems, it is not a significant event as long as the enterprise has the ability to give adequate response to this event: i.e. it has the operational and management capabilities to deal with the consequences and no change in the management or operational abilities of the enterprise are needed.
On the other hand, a political change in the legislation in one of the main markets of an enterprise may be treated as very significant if e.g. it changes the conditions under which the company's products can be sold in that country. Such an event is significant, because if no adjustment is made in the management or operation of the company to address the above-mentioned environmental change, the company may eventually lose that market. As a consequence, some production facilities may have to be moved to the country which is the main market of the company, or new material technology should be introduced to replace existing one (to be phased out due to international agreements).
The above two examples describe external events that may create a need for change. Many internal events can be responsible for the same, such as the appointment of a new person to a key position and the likely change in company policies, creating the opportunity to do some things better and differently, or stop some current practices from being followed. If the consequence is that the usual management and operational practices and processes can no longer be followed, this event can trigger an important change process in the company.
In the case of project enterprises, since they have a limited life span, change initiatives are either handled by the project, or are handled in the decommissioning phase that documents problems that new project designs can take into account to improve future project performance. In project enterprises, if a problem arises on the lower level of project operations, the first person to know about it is often the project manager (PM). The PM looks at where this information comes from, what events happen in the area, and considers if the PM is the right person to address the problem, as well as whether it is the given project that needs to handle it. However, it is often realised that the root of the problem may lie on a higher level, such as in the business enterprise that created the project.
It is useful to state the interpretation of the event as a description of the problem it demonstrates, as well as its ongoing consequences - but without any reference to the solution. If the manager recognising the event is not in power

to solve it then agreement is reached easier on the existence of the problem if the interpretation does not immediately jump to a conclusion regarding the necessary action. Finding the right level of action is a process of negotiation and persuasion, otherwise information may not be followed by action.

In summary, *significant* events are those having consequences that the company cannot deal with using its present management and operational capabilities. Because so many events happen in the enterprise, there must be a filtering process that identifies those events that are worthy of attention and thus become 'significant'. For the enterprise to do such filtering there must be suitable indicators available and an information gathering processes in place that are able to alert management of the potential need for change.

There may be a number of problems with a company's ability to get alerted to significant events.

- While each manager has a set of objectives (which is a mixture of company and private objectives) and is alerted to events that violate these objectives, there are many indicators that do not get to the attention of managers because of the lack of properly set up information / communication channels (whether through human communication or through a Management Information System). Thus not all information that should be interpreted as significant gets to the attention of management and therefore remains unnoticed.
- Not every manager has the necessary competency or the time to interpret information correctly.
- Events may be recognised as significant, but the conceivable response from the perspective of the individual manager may cause a conflict between the company's system of objectives and the individual's private objectives.
- The event may be recognised, but the conceivable response causes conflict between the company's set of objectives and the expectations of the external world, and thus the manager may have the tendency to ignore the relevant information in order to avoid conflict.
- A manager may recognise the significance of new information but is unable to communicate this convincingly to other managers, and so the information 'dies off'.

As seen from the above discussion, the use of the 'significant event' concept is intricately dependent on the personality, communication and leadership traits of managers as well as the company's communication culture, and should not be thought of as a mechanistic 'alert system' that solves the problems of timely response on behalf of the company. The ability to perform the actions described below critically depend on the existing ability of company managers to communicate the need to act on new information, and often requires the use of informal or formal but not instituted communication channels to establish a joint wish to act.

8.4 Organisational Performance Assessment

Organisational performance in light of these significant events may be assessed formally or informally. Organisational performance may be assessed before a change is initiated, i.e. for hypothetical future scenarios, using

- peer appraisal (subjective), or
- cost/benefits analysis (objective)

or after the change has been completed, i.e. for actual past events, using

- bibliometric measures (objective), or
- technical review (subjective).

These measures are valid at each end of the scale, but tools to assess ongoing activities and performance have not been well developed except for financial indicators and market analysis. Using the results from sales or marketing departments for assessment of the enterprise is a post-event analysis. In reacting to senior management's need for strategic direction, competitive intelligence and marketing professionals are often pulled in a number of directions gauging which direction is best for their enterprise. The assessment should not only consider the company as a whole (strategic thinking), but should include respective business units (or business entities) as well.

Identification of problems or strategic opportunities in the enterprise results from knowing the parts that comprise the enterprise. When executive management identifies a strategic opportunity, it is often done with explicit as well as tacit understanding of the abilities of the company or sections involved. The strategy concept is filtered to the next level of management, which will assess whether the strategy is feasible or if refinement is required. Planners then help the strategy makers to identify players and business entities and give feedback on which sections of the organisation will be utilised to enact the vision. Companies who monitor their performance continuously rather then only at selected milestones are better prepared to make strategic decisions in time, because such monitoring positions them better to *recognise the time* when change initiatives must be launched. In recent years (1990s) this has been well recognised and put into practice in many enterprises, through the development of a system of *performance indicators* that allow an ongoing assessment performance.

It is noteworthy that while financial and market indicators have been in use for long, these indicators are usually reflecting the past or recent past performance of the company (called *lagging* indicators), thus are of less use to support strategy making then indicators that characterise the company's potentials for future performance (*leading* indicators – as discussed in Chapter 4).

Balanced scorecards and the more elaborate European Foundation for Quality Management (EFQM , 1992) model (Chapter 4) allow such indicators to be taken into account. These leading performance indicators asses the company from the point of view of its ability to perform well in the future.

Therefore, should the company not have a system of leading performance indicators in place, a strategic initiative to develop them is itself an important change project.

8.5 Determining the Scope of Change in Enterprise Entities Involved in the Company's Business Model - What Enterprise Entities Will Be Affected ?

It may not have been entirely clear when it was previously said that significant events are those with known or potential consequences on the operation of 'the' enterprise, because in fact several Enterprise Business Entities may be affected by a significant event and thus a change initiative.
To be able to locate the extent and nature of change, the initiating sponsor and the champion of the change initiative need to identify the enterprise business entities involved in the change and the relationships between these EBEs.
If the change initiative is about a new business opportunity, some of the EBEs may be non-existent, some may exist with little or no change required, and some will require change to implement the initiative.
The system of relationships between EBEs involved and their relationships (the way they are, or will be, doing business together) is called the 'Business Model' (a possible representation of which is shown in Fig. 8.1).
Some change initiatives require the current Business Model to be changed, while some only change one or more aspects of a given EBE (such as improvement of processes, installation of new equipment or software systems, etc). If the change initiative is to achieve Enterprise Integration (integrated information and material flow) , the relationships between several EBEs is usually affected, resulting in changes to the Business Model.

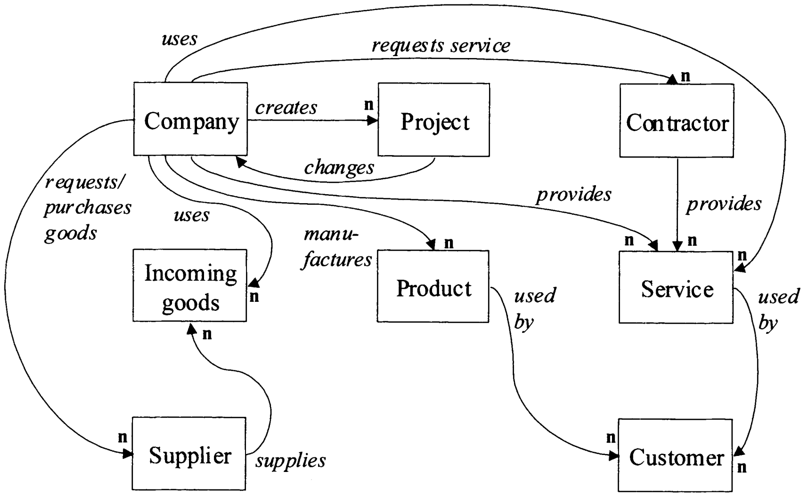


Fig. 8.1. Representation of the Business Model using a simplified Entity Relationship diagram describing the roles of a company, suppliers, contractors, projects and the product

A somewhat different representation of the business model is a 'chocolate bar' diagram, in which each EBE is visualised through a graphical symbol (resembling a chocolate bar, as shown in Fig. 8.2) representing the life cycle of the respective EBE and use arrows to represent the contribution of each EBE to the activities of the other, as shown in Fig. 8.3 (also refer to Fig. 2.4, Chapter 2).

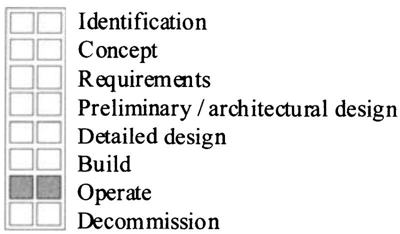


Fig. 8.2. A 'chocolate bar' representing the life-cycle of an enterprise entity

The advantage of this representation is that the role of each entity is defined in more detail and most importantly, the role of an entity in the life cycle of

the other entity is identified. This is naturally an advantage when considering to develop a road-map for a change process.

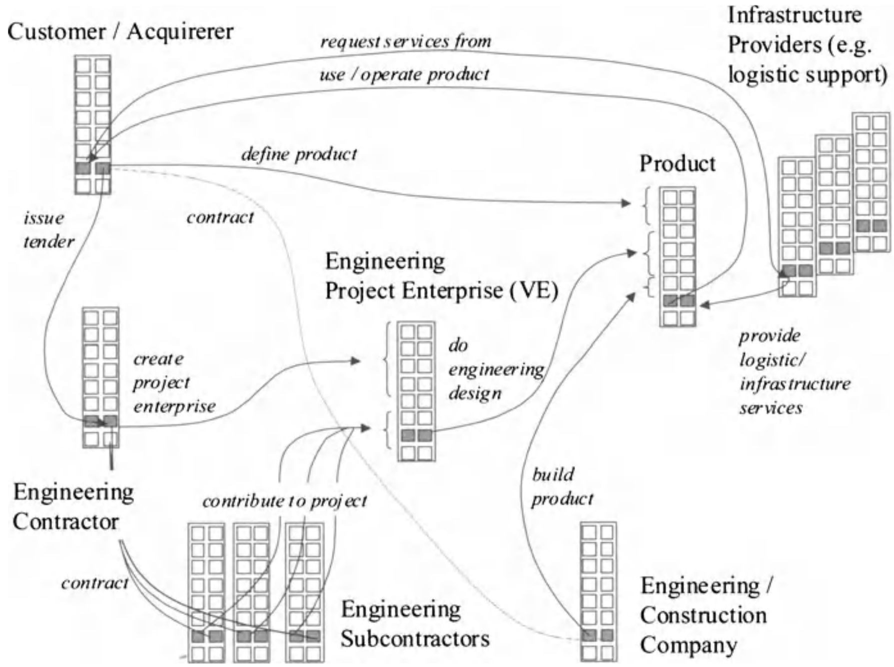


Fig. 8.3. Business Model representation using a chocolate bar diagram [4]

Additional diagrams (models) may describe other relevant aspects, such as the value chain (how EBEs contribute value), the supply chain (the material and service flows[5]), and the investment flow (how the flow of investment produces returns).

If the change initiative does not change the way business is done in the enterprise, then such a chocolate bar diagram may be sufficient and can be used as a map to point out the EBEs and the locations in those EBEs where the initiative is likely to cause local change – e.g. a change initiative may be found to influence manufacturing policies and procedures of an EBE, thus the manufacturing facility's policies and procedures need to be looked at (i.e. a concept level change may be necessary, with on-flowing effects on their implementation).

However, if the change initiative substantially changes the way business is done in the enterprise, several alternative decompositions of the enterprise into constituent EBEs could exist. The decomposition to be followed is not

[4] example shown from the one-of-a-kind product (OKP) industry

[5] e.g. using a functional model or process model (refer to Chapter 11)

determined by the initiative; hence, some analysis is necessary to determine the one to be selected. The analysis involves reasoning and evaluating from a number of points of view, such as politics, technology, marketing, knowledge production and retention, viability of each EBE as a separate entity, finances, human aspects, risk, etc. All in all, the change initiative's *feasibility* needs to be studied so as to make an informed decision.
While the invention of new business ideas – such as developing a new business model – is a creative process, several techniques exist to supply important information to the designer and implementor of these new ideas in order to establish the feasibility and adjust the idea, so as it may become a viable and attractive proposition.
The change identification process should not take a long time (techniques to speed it up are presented below), but without it one would expect more surprises in the future when the change process is in an advanced stage. Investing in 'as-is' analysis is appropriate when considering to use or reuse elements of the company through improvements, rather then planning a radical change or when the necessary improvements are not evident.

8.6 Foundations of Change

Initial phases of any change management process are enhanced if the process has solid foundations. Such foundations may be built through the study of:

- whether and why the change is necessary,
- what is to change,
- how, when and where change will occur,
- who can effect that change, and
- what are the potential effects of such change (Mink et al. , 1993; Handy , 1985; Jick , 1995; Zachman , 1987).

Resulting from the identification that change can benefit an enterprise or that change is necessary for the enterprise to retain its competitive advantage, the choice of the transition method is the one management often finds to be the most difficult. It is this choice which often determines the future success of an enterprise.
The understanding of the selection process of a change methodology for an enterprise within any industry sector requires the consideration of a number of concerns in two areas.
Firstly, one must consider

- the target enterprise as a provider of services or producer of goods and how changes may affect the continued provision of these,
- the necessity for change, and
- the strategic management process of such change (e.g. (Jick , 1995)).

Secondly, one must consider the tacit notions, concepts, requirements and ideals of the change methodology – whether they are acceptable for the type and style of people who work in the respective industry sector.
When confronted with a methodology for enterprise assessment or enterprise change (in the case of an existing enterprise), there are many aspects that determine its acceptance by enterprise- and change management stakeholders. These aspects pertain to the enterprise stakeholders, the activities and processes producing the enterprise product or service, the activities and processes managing and controlling the enterprise, external enterprise relations, the reason for existence, etc.
Every enterprise change methodology, regardless of whether it is for a new- ('green-field') or existing enterprise rely on an understanding of current best-practices from industry and / or academia – at least those aspects of current best-practices one wishes to (or must) incorporate into the new- or renewed enterprise. These aspects are not limited to business processes, but include cultural norms of the given business environment and of the organisation (is the enterprise mature enough to handle formal modelling of its processes and information processed?), how the methodology fits into current and future strategies (does the methodology require reusable skills?), and current and projected use of technology for the production of goods and delivery of services (does the methodology come with supporting tools for the design and implementation of new technology?).
In general, apart from the functional appropriateness of the methodology it is of important consideration whether the technology and skills needed to adopt the methodology are acceptable for the organisation / by people in the organisation.

8.7 The Function of the Identification Activity

In changing state from one form to another, or in creating a new enterprise, the functions of the identification activity are as follows.
The identification phase *helps direct the intended change process*, whereby a mission for the new (changed) enterprise is established. Either the initiating sponsor, or a project team performing the feasibility study can carry out this identification and (mission) direction of the change process (refer to Chapter 2), but in any case consensus must be achieved between the stakeholders.
The identification phase must *propose a decision* on how to distribute and coordinate the future business management and production (service delivery) systems; often this co-ordination must be done on a global scale. Especially when defining the set of required integration services which "need to be widely supported to facilitate interoperation in a flexible way between reusable system components" (Weston et al. , 1998, p.5), it is critical to identify the EBEs together with their *relations* to other internal or external EBEs because the

standardisation of an integrating infrastructure is a critical success factor of the integration process.

Organisations have both history and memory, which organisational stakeholders carry in their heads, basically "shared understandings and mental models that have accrued over time" (Boudreau and Robey , 1996, p.45). The identification process attempts to uncover any such *factors* that result from this background and may *influence the success of the change process.*

Decision needs to be made on the intended level of technical maturity of the future organisation ('technological vision'). To plan for the use of the most advanced information technology is possible but if a complete changeover is unlikely to happen shortly (e.g. because of restricted availability of funds) the vision must take into account the limitations of the present technological infrastructure. "Where advanced information technologies are advocated (as they usually are), few organisations seem prepared to finance the mass migration towards an unproven future. Why should new processes be designed without incorporating the assumptions affecting their implementation? The assumption that design and implementation are activities that can be separated is another fallacy" (Duimering et al. , 1993). If the present situation is largely ignored then designs based on the vision may easily become infeasible and need to be modified on implementation to account for pre-existing conditions. These conditions should be taken into account in the initial life-cycle phases, certainly earlier then detailed design (Boudreau and Robey , 1996, p.45).

8.8 Assessing the Present Situation and the Feasibility of the Intended Change

At this point, either (a) there is an identified *need for change* in the enterprise, with the scope and focus of change having been identified by the project champion or initiating sponsor of the change process; or (b) there is an identified *need for assessment* of the enterprise or its domains of activity.

The main thrust of this section is looking at whether an enterprise change, a strategy or a vision is practical. The aim is to assess the feasibility of the strategy, what is necessary and sufficient to be done for its implementation and where the enterprise stands now relative to where it wants to go, as well as which standards exist in the world to aid in the assessment and also in the subsequent stages.

To ensure quality systems / products / service delivery, in time, on budget, it is advisable to have standards rigorously implemented in the organisation and the development process itself.

What previously required risky investments, such as outsourcing development, can now be achieved through collaboration. For some enterprises this will be the only realistic way to leverage organisational assets and will become a necessity that should be appreciated sooner, rather than later. Standards will

help the process get there faster, allow the use of a proven way of doing things and ensuring that minimal mistakes are made on the way. Standards can assist in executing corporate and business level strategies. Standards present a powerful way to avoid common pitfalls – after all, they are put together after significant inputs from relevant entities and developed by domain experts.
There is a reason why much of the focus so far has been on the current as-is or self-assessment. No matter how radical the change, one must understand the business functions performed.

- The AS-IS model helps define the current position of the enterprise ("if you don't know where you are, a map won't help");
- The Enterprise Models define a destination.("if you don't know where you are going, any map will do");
- Enterprise Modelling provides a reasonable way of validating the maps ("if your map is seriously flawed you are in deep trouble").

It is important to understand what infrastructure exists for new business initiatives and what infrastructure needs to be built to support those initiatives. The analysis or self-assessment uses two viewpoints:

- *environmental appraisal*, looking at the enterprise business entity from the outside – how it relates to its environment, what is its place (Section 8.8.1), and
- *Corporate appraisal*, looking at the enterprise business entity as an autonomous entity and relating it to the requirements and challenges that the outside word places upon it (Section 8.8.2).

Each of these analyses may be carried out from several perspectives, these perspectives have a level of maturity associated with them, and the level of maturity has an associated standard or current best practice. The necessary capability / maturity levels for the future state have to be known to be able to define a feasible strategy, as they can be used to assess how the EBE's existing resources will respond to the change, i.e. whether the vision is achievable.

8.8.1 Environmental Appraisal

The main purpose of this activity is to pinpoint the current environmental influences acting upon the EBE. The evaluation of each of these environmental factors should include a statement of the present as well as expected future trends which assist in the guidance of the development of future strategies and visions.
The environmental appraisal:

1. Reviews the market in which the EBE operates (is it growing, shrinking, changing/shifting, offering new opportunities, new competitive situations, what are the present and future target markets etc.);

2. Reviews the competitive forces impacting on the EBE. For the change initiative to succeed, management must have a thorough understanding of the EBE's competitive environment. It is the nature and structure of the chosen industry (and how the organisation deals with these) that indirectly determine the profitability of the organisation. Profitability is strongly influenced by how the organisation operates within the five competitive forces (Porter , 1980, 1985);
3. Reviews the current use of technology within the EBE[6] relative to technology use in the competitive environment to identify deficiencies, need for modernisation, or expected future need for change in this regard (Is the present technological capability and capacity adequate? For how long? Is the required capability and capacity likely to change and how?);
4. Conducts an economic analysis to ascertain the performance position of the EBE within the economic environment (What is the present return on investment? What value adding activities does the company do and which activities add little value or add them at a low profit margin? Does the company offer competitively priced products and services for the target markets?);
5. Conducts a government (political) analysis to show the current legislative and taxation constraints on the EBE (Are there any present or changing legislative or taxation constraints that prevent the present operation from functioning effectively and efficiently? Are there any anticipated changes in these?).
6. Conducts a social analysis of current social expectations impacting the EBE(s)[7]; human resources, work-force availability and culture (Are there any issues regarding the present job market? Is there a problem with employee retention and employee satisfaction? Are there any union concerns? How does the company handle knowledge retention? etc).

8.8.2 Corporate Appraisal

The purpose of a corporate appraisal is to investigate the internal factors affecting the current performance of the EBE. The purpose of this activity

[6] what types of technology the EBE is currently using (specific types of food technology, information technology, etc). For example, in the Hospitality Industry, it is a substantial aspect of technology that it is employed to facilitate efficient and effective customer-employee interaction in attempting to raise the quality of service (Olsen , 1992)

[7] e.g. according to the host country and client culture. The social expectations are intrinsically important in the pre-selection process that clients conduct in the comparison of enterprises whose services they intend to use. Knowledge of host country and client culture assists in appraising the needs and expectations of present and future customers, which facilitates the development of a good image of the organisation. Continual maintenance of this notion of image will assist the organisation in achieving its objectives

can be summarised as identifying the domains of activity of the EBE, what processes are important within these domains, what is the current situation within the domains of activity, and finally, contrasting the desired / changed mission, vision and values with the findings.
The first part of this activity includes an identification of the EBE's current domains of activity. The second part is a review of these domains, with the view of now finding concrete objectives for change in the investigated domains. Thus the effects of the new strategy are assessed.
The corporate appraisal:

1. Defines the boundaries of operation of the EBE's domains of activity, delineating the tangible and intangible links between the EBE under investigation and other EBEs as well as among the domains identified (i.e. points of interaction, existing relationships and any significant influences deemed important). For each domain, its business processes should also be listed at a high level. For example, a change in the way customer service is performed may need the introduction of corporate-wide procedures, and thus impact on tradition as different geographic areas may have their own way of servicing customers. Standardisation of procedures in this regard suddenly creates links between corporate management and branch management and impact on the autonomy of branches. The list of processes (tasks) that customer service performs may, for example, include basic technical support, handling warranty-related requests, and organising maintenance and spare parts service, as well as area marketing. Other connected domains may be specialist technical service (provided by a corporate domain), marketing (corporate) and spare part manufacturing (corporate or geographic area), with links to the branch customer support domain.
2. Identifies the business goals and critical success factors for the EBE and its domains. Identified for each domain are the internal or external competitive forces impacting upon it, and the competitive strength the domain currently has, as well as the competitive strength the domain should have. E.g., the business goal for the branch customer support may have been derived from corporate goals, satisfying a corporate strategy, such as optimising customer support in terms of quality and uniformity of service, while at the same time, through knowledge sharing, decreasing cost. The derived goal for the branch may be to be able to provide one point of contact service for each product and the ability to organise and train local services as an extension of the company's operation, as well as to host processes that include corporate services (such as may be needed in case of problem escalation). Additional goals may be to improve the gathering of technical and non-technical information for product development and of marketing information for corporate market planning.
3. Assess the EBE through the characterisation of each domain and, separately, each business process within these domains (Rolstadas , 1995;

Bussler , 1995). The result can be represented in a table that states the most important observations regarding the current state of the EBE and its domains, showing *opinions* regarding:

a. The decision / management structure. For example, is the management structure deemed to be highly developed and capable of dealing with the necessary change, as well as with the new mode of operation after the change? Is the management structure generally adequate, but some adjustment is needed? Does the assessment conclude that an entirely new management structure seems to be necessary? Is the maturity level of management – in terms of the EFQM model (EFQM , 1992) adequate, needs small adjustment or major revamp?);
b. Information and material flows (i.e. their interfaces, control and other relevant properties). For example, is the information flow in the EBE(s) integrated and well defined, or does it need improvement – as opposed to the information flow not being defined at present to the level where small improvements are likely to solve the inadequacies under the new business model? Is the present level of definition of the information flow inadequate to the extent that a completely new design is needed?;
c. Economic performance. Are present processes of the value chain adequate, or small improvement are needed, as opposed to perceiving a major revamp of processes to achieve drastic improvements in the value adding process of the company? In a branch service domain, is the cost of providing the service[8] acceptable? Is the speed of service adequate?;
d. Human resource functionality[9] Is the human resource adequate, and gradual training or education or small adjustments in the workforce is likely to be able to support the change? Are there are major inadequacies in the allocation of human resource? Are the competencies, skill and knowledge level of the workforce adequate?;
e. Level of maturity of the domain. Regarding the maturity of processes: are processes repeatable, well defined, or optimising? How does the existing level of maturity contrast with the needs of the new business model? E.g. if the company has repeatable processes, but not defined explicitly, this may prevent the company from entering into alliances with other companies as may be required by the new business model outlined in the strategy.
f. Other problematic areas.

---

[8] note that cost is a derived property of the processes followed, and as may be expected, the change may effect cost by either decreasing or increasing it

[9] it is important for management to see whether the EBE has all the necessary business functions present that should be contributing to its products or services. Any functions that are missing or do not perform to the level needed, and functions that are performed but do not add value need to be identified.

The analysis or self-assessment should not be limited in its considerations, but analyse to a pragmatic level only those aspects which are conducive to the EBE's strategy and Business Plan as well as concrete goals. The observations may span each or any aspect of the EBE, its domains, and business processes such as listed in the example shown in Fig. 8.4 or in the following subsections.

EBE Domains of Activity (generic) / Perspective	Accounting, Financial & Economic Mgmnt	Research & Development	Product Development	Technological Development	Internal Services / Administration	Human Resources Mgmnt	Knowledge Mgmnt	Quality Mgmnt	Project Mgmnt
Leadership*	1								
Decisional / Mgmnt Structure	0								
Strategic Planning	?								
Strategic Mgmnt	?								
Client / Customer and market focus	1								
Product and Service Processes	1								
Info & Material Flows	?								
Business Processes	?								
Support Processes	1								
Economic Performance	?								
Human Resource Functionality	1								
Public responsibility / social awareness	1								
Customer relationship and satisfaction	?								
Employee Education, Training, and Development	?								
Employee Well-Being and Satisfaction	?								
Cultural influence	1								

* Clear and visible direction, values and expectations

Fig. 8.4. EBE Characterisation

The characterisation of each domain or EBE shows whether the as-is situation is clearly understood and results in the decision on where to (and not to) undertake a more thorough assessment. It is likely that such a rapid assessment creates a map that guides the attention to areas where information is missing to support decisions thus may need an as-is analysis. Therefore, such assessment must be performed by insightful managers without going into a lengthy analysis process. Once the need for analysis of certain domains from a certain aspect is established, an analysis may be commissioned. It is expected that in any relatively healthy company only a fraction of the above table will signal the need for analysis.
Note that an as-is analysis may have two objectives:

- To obtain a model reflecting the current state of affairs in the analysed enterprise domain, resulting in a deeper understanding of how the enterprise is operating, and is to be conducted in order to identify concrete improvement needs; and
- To develop expertise in how to use enterprise modelling for the analysis of the business domains in question. Often, the first step in building enterprise architecture capability is to engage in smaller projects that establish enterprise modelling skill and knowledge, because without it management will have difficulty in developing new to-be designs of the required quality (refer to 11).

If the analysts are inexperienced in modelling and analysis, then the second gain alone is worth the effort to prepare for the next step, which is the implementation of the new strategy. As a result, the analysts will see the value of applying analysis and modelling in the subsequent design and planning process. They will also gain some proficiency in applying analysis methods to the particular business domain.
It may happen that the analysts realise the current domain is in such a bad state (**'0'** in the table in Fig. 8.4) that more as-is analysis is not likely to contribute to any improvement. In this case, the as-is analysis may be unnecessary but the opportunity of training on the as-is analysis must be replaced by another form of training (e.g. through the study of existing textbook models, for example). If in the given domain the analysts expect improvements to be achieved through continuous improvement, and models of the current domain are not available, or if the rapid assessment is uncertain in the characterisation of the given domain (**'?'** in the table in Fig. 8.4), an as-is analysis will contribute to the assessment of the feasibility of the change, as well as will contribute concrete change objectives to the planned change. If the domain is judged to be performing to the satisfaction of all parties (**'1'** in the table in Fig. 8.4) then from the feasibility point of view there is no need for as-is analysis of that domain and the domain is suitable for inclusion in the new strategy. Still, since satisfactorily performing domains have interfaces with other domains of activity, the change initiative may need some corrective action at the interfaces of the existing domain. However, this becomes apparent

only in light of architectural (preliminary) design. As a result, management may or may not mandate that interfaces of such domains remain unchanged – in the lack of such mandate, architectural design may work with a greater degree of freedom.
Due to the cultural aspects of change, not mandating such a constraint may cause difficulty in accepting the change, because people working in that domain will be hard to convince of the need to introduce any change. Thus, such lack of constraints must be justified in light of the human factors in the company.
Every such as-is analysis results in a representation of the domain which is then assessed in its own right (e.g. are there any structural problems, missing links, qualitative or quantitative deficiencies etc.?) and compared to other similar domains of other enterprises, using benchmarking techniques (e.g. (Lacity and Hirschheim , 1995)) as will be explained later in this chapter.
Below is presented a set of possible techniques applicable *if* as-is analysis was found necessary.

8.8.2.1 Review of Decisional / Management Structure

To complete this type of as-is analysis, the analysts should review the decision and management structure (control hierarchies) of the EBE and its domains for which an as-is analysis was deemed necessary. The analysis will identify the existing chains of authority and spans of control, which affect the communication and information channels within and between the domains and identify the decisional centres and decisional roles within the domains as well as their connections.
A GRAI-grid (Doumeingts et al. (1996); MacIntosh and Carrie (1995) and Bernus et al. (1996)) may be used in this analysis (refer also to Chapter 12). Based on this assessment, the analysts can 'map' the current EBE's organisation onto a GRAI-grid, labelling the EBE's organisational units (people / groups) with 'roles' on this grid. This analysis normally allows for the identification of problem points: e.g. the analysts can, through this analysis, identify non-contributing roles, conflicting roles, inconsistent decisional frameworks, missing decision roles, conflicting roles, etc. Analysing the decision model of the present EBEs (domain of activity) will also allow the analysts to answer part of the questions of the People Capability Maturity Model (PCMM) (Curtis et al. , 2002). A PCMM is a model describing various levels of organisational maturity regarding how it manages its human resources. Further answers to the PCMM can be derived from the functionality analysis below and from the environmental analysis.

8.8.2.2 Review of Information and Material Flows

This analysis identifies the information and material flows within- and between the domains of activity (as well as among processes in this domain), in order to show their interfaces.

There are two typical problem types that can be pinned down by this analysis:

- The interfaces (information and material flow) within the analysed EBE and domain are not well defined (e.g. information and material flow are not coordinated, the data and material exchange is based on ad hoc methods, etc.);
- The interfaces are defined, but data and material exchange is inefficient (e.g. due to too many process steps, bottlenecks, inefficient technology, etc.).

The identification of these problems can be done through:

- Qualitative information and material flow models of the domain (e.g. IDEF0, Data Flow Diagrams, CIMOSA functions and information views, Petri Nets, Entity Relationship- or Object-Oriented schemata);
- Quantitative analysis (mostly achievable through simulation modelling) to characterise the information and material flow in the domain (e.g. using CIMOSA, Timed Petri Nets, etc.).

8.8.2.3 Review of Economic Performance

The analysts review the economic performance of each domain of activity for which an as-is analysis was deemed necessary. The analysts should identify if there exists performance indicators and corresponding criteria to measure the performance of the domain. If no such criteria exist, then the analysts must develop performance criteria to measure the performance of the domain. The performance criteria should be evaluated to assess their validity as they are open for customisation according to the domain; i.e. not all performance indicators / criteria are applicable in all domains. If the performance criteria are not being met, it indicates that the domain may require change.
A useful way of deriving economic performance indicators (and identifying performance drivers) is for the analysts to create an Activity Based Costing (ABC) model of the investigated domain or process (Lucertini et al. , 1995). The ABC model could either be separately created or as an extension to function and resource models (e.g. CIMOSA or IDEF0) obtained as part of the information and material flow analysis.
The development of a new performance measurement system is of course not the task of the analysts; it could be done by the planning team as part of the change program.
To summarise: either there are performance indicators which can be used to benchmark the EBE against standards, competitors and expected trends (in which case benchmarking is used to assess the need for change), or there are no performance indicators (or there exist only partially developed ones) in which case the need to develop these is part of the task of the ensuing planning process.

8.8.2.4 Review of Human Resource Functionality

The human resource functionality is reviewed from the point of view of the human resource structure, function and roles (Curtis et al. , 2002) showing how the domain operates at the level of the organisational actor – i.e. the individual in the organisation – as there may be a number of human-centred production and service processes. Note that even though a similar type of analysis has been carried out for human management roles in the first step (decisional model), this analysis looks at the individual in the service / production process.
In this step, the analysts should consider holistic aspects of the human resource, as it often happens that unexplained (i.e. not modelled or quantified) characteristics of the human resource influence the service / production. Personality and management intuition should not be disregarded in a misguided quest for quantifying every decision. Should the quantified aspects and the intuitive aspects of the analysis be in conflict, this activity will shed light on problems that need to be confronted.

8.8.2.5 Summary of As-is Analysis

It is expected that these analyses will fill in the gaps in the understanding of the current as-is situation within the EBE (there will be no question marks - '?' in the table in Fig. 8.4).
The findings from these investigations and reviews can now be contrasted with the currently perceived mission, vision and values of the EBE[10] in order to identify those processes which do not contribute towards competitive advantage, or the performance of which is not at the desired level. The analysts also identify those domains of business activity which should have clearly defined performance indicators, but do not currently have them.
On the corporate appraisal level it is desirable to identify not only the problem performance indicators but also the problem performance *drivers*. This is because it is those domains of activity (or processes within those domains) that need change which encompass the performance drivers at fault.
In those enterprises which already have well defined processes, the corporate appraisal could unveil the processes with the performance drivers that cause concern. Thus, conducting an analysis here has the potential to identify a relatively smaller area that needs change. It should be kept in mind that significant performance drivers may be outside of the given domain, or even the company[11].
However, the enterprise may not be at the level of maturity of 'operating well defined processes' (Paulk et al. , 1993) in which case a too deep analysis of the present situation may not promise much, and the change should be directed to defining processes as a priority in the change process.

[10] if applicable.
[11] this indicates strategic instability for the enterprise.

A valid reason for undertaking this analysis may be to bring the EBE from one level of maturity to the next (Paulk et al. , 1993). It will be then up to the planning team to ensure that the performance drivers with significant impact on performance indicators of the EBE become controllable in the to-be state of the EBE. Note that a number of Capability Maturity Models have been developed and were recently integrated into a model that covers systems engineering, software engineering, integrated product & process development and supplier sourcing (SEI , 2002).
This allows not only the identification of potential areas of improvement, but also the ability to recognise non-value-adding processes (Sager , 1990). Note that the emphasis here is to recognise any problems, contradictions, missing details etc., *not* to solve them. Importantly though, it is not advised to take a dogmatic view of value adding; value can be added in form of the product or in form of creating *future capability*, where the consumer of the product is the future enterprise.
This subsection has given details of the corporate appraisal. The result of this appraisal allows the analysts to clearly see whether:

- there is any strategic gap between the current practices and the perceived current operational mission, vision and values (Howe , 1986);
- the applicability and maturity level of the domains of operation within the organisation;
- the suitability of these domains to new strategic initiatives, and
- the aid that standards can give in the change process.

The result will then be used to direct the change within the EBE.

8.8.3 Assessing Against Others – Benchmarking

Benchmarking (as part of the identification process) is a technique that is employed to assess the current position of the enterprise in terms of measurement for its own baseline, or in terms of measurement against leaders in the industry sector. However, the current position of the enterprise in this respect may not be indicative of its likely future position. Therefore, enterprises may use capability assessment methods which measure the enterprise against a broader set of criteria (perhaps not attained by any other enterprise yet).
Enterprises may thus identify some universal need for improvement and institute timely change. Benchmarks include the assessment of the enterprise's production and service delivery processes, and the quality of enterprise management. Since the ability of quick reaction to change in the environment is such an important property, measuring in some way the ability of the enterprise to quickly adapt to new business environments, e.g. by using so-called *agility metrics* (Goranson , 1998) is a promising approach.
Regarding capability assessment, one can use standard models that describe hypothetical states of maturity of the enterprise and establish metrics so as to

be able to assess the level of maturity that the organisation is at. It is generally accepted that maturity levels are a natural progression of enterprise evolution, but of course there is nothing to say that other innovative, non-evolutionary changes could not provide better alternatives for the future.

References

Bernus, P., Nemes, L. & Williams, T.J. (Eds.) (1996) Architectures for Enterprise Integration. London : Chapman and Hall

Boudreau, M. C. & Robey, D. (1996) Coping with Contradictions in Business Process Engineering. In *Information, Technology and People* 9(4) MCB University Press. pp40-57

Curtis, B., Hefley, W. E., & Miller, S. (2002) The People Capability Maturity Model – Guidelines for Improving the Workforce Boston : Addison-Wesley

Doumeingts, G., Kleinhans, S., Clave, F., Malhene, N. Prevot, B., Ohrensstein, H. & Rojas, H. (1996) Improving the Bid Preparation Process through a Benchmarking Approach: an Industrial case study in 3 French companies (among which 2 SMEs). In *Modelling Techniques, Business Process Engineering (BPR) and Benchmarking*, International Workshop Proceedings, IFIP WG 5.7, April 18^{th} - 19^{th}, 1996, Bordeaux, France

Duimering, P. R., Safayeni, F. & Purdy, L. (1993) Integrated Manufacturing: Redesign the Organization before Implementing Flexible Technology. *Sloan Management Review*, Summer, 1993, pp47-56

EFQM Excellence Model (1992) Bruxelles : European Foundation for Quality Management. http://www.efqm.org

Pfeiffer, J. W. (1991) Strategic Planning: Selected Readings. San Diego : Pfeiffer & Co

Goranson, T. (1998) Architectural Requirements of Commercial Products. in Bernus, P., Mertins, K. and Schmidt, G. (Eds.) Handbook on Architectures of Information Systems, Springer Verlag, pp711-732.

Hammer, M. & Champy, J. (1993) Reengineering the Corporation: a manifesto for business revolution. NSW, Australia : Allen and Unwin

Handy, C. B. (1985) Understanding Organisations. London : Penguin Books

Howe, W. S. (1986) Corporate Strategy. London : Macmillan Education

Jick, T. (1995) Accelerating change for competitive advantage. Organizational Dynamics, 24, pp77-82.

Kofman, F. & Senge, P. (1993) Communities of Commitment: The Heart of Learning Organisation. In *Organizational Dynamics*, 22(2) pp5-23

Lacity, M. C. & Hirschheim, R. (1995) Beyond the Information Systems Outsourcing Bandwagon. New York : John Wiley & Sons

Lucertini, M., Nicilo, F., & Telmon, D. (1995) Integration of Benchmarking and Benchmarking of Integration. In *International Journal of Production Economics: Manufacturing Systems, Strategy And Design.*, 38 (1)., March 1995. pp. 59-71

MacIntosh, R. & Carrie, A. S. (1995) The Use of the GRAI Method in Re-Engineering. In Browne, J. & O'Sullivan, D (Eds.) *Re-Engineering the Enterprise*, London Chapman and Hall. pp273-283.

Mink, O., Esterhuysen, P., Mink, B. & Owen, K. (1993) Change at work. San Francisco : Jossey Bass

Olsen, M. D., Tse, E. C. & West J. J. (1992) Strategic Management in the Hospitality Industry, New York : Van Nostrand Reinhold

Paulk M.C., Curtis, W., Chrissis, M. B., & Weber, C. V. (1993) Capability Maturity Model, Version 1.1. *IEEE Software*, 10(4) July 1993. pp18-27.

SEI (2002) CMMISM for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Version 1.1 (CMMI-SE/SW/IPPD/SS). Technical Report CMU/SEI-2002-TR-011, 1 March, 2002. Pittsburgh, PA : Software Engineering Institute, Carnegie Mellon University.

Porter, M. (1980) Competitive Strategy: Techniques for Analysing Industries and Competitors. New York : Free Press

Porter, M. (1985) Competitive Advantage. New York : Free Press,.

Rolstadas, A (1995) Self- Assessment - A Tool for Business Process Re-Engineering. In Browne, J. & O'Sullivan, D. (Eds.) *Re-Engineering the Enterprise*, London Chapman and Hall. pp175-186

Bussler, C. (1995) Workflow-Management-Systems as Enterprise Engineering Tools. In Bernus, P. and Nemes, L. (Eds.) *Proceedings of the Working Conference on Modelling and Methodologies for Enterprise Integration*, London Chapman and Hall. pp234-237

Sager, M. (1990) Managing Advanced Information Systems: An Introduction to Frameworks and Experience. Brookvale, NSW, Prentice Hall

Spector, B. A. (1989) From Bogged Down to Fired Up: Inspiring Organizational Change. *Sloan Management Review*, Summer, 1989, pp29-34.

Williams, T.J., Rathwell, G.A., Li, H. (Eds) (1996) A handbook on master planning and implementation for enterprise integration programs (based on the Purdue enterprise reference architecture and the Purdue methodology) Report Number 160, West Lafayette, IN : West Lafayette, Indiana : Institute for Interdisciplinary Engineering Studies, Purdue Laboratory for Applied Industrial Control, Purdue University

Weston, R. H., Coutts, I. A. & Clements, P. E. (1998) Integration Infrastructures for Agile Manufacturing Systems. in Bernus, P., Mertins, K. and Schmidt, G. (Eds.) Handbook on Architectures of Information Systems, Berlin : Springer Verlag. pp733-764

Zachman, J. A. (1987) A Framework for Information Systems Architecture. IBM Systems Journal, 26(3).

Part III

Defining the Requirements for Enterprise Change

9

DEVELOPING THE ENTERPRISE CONCEPT - THE BUSINESS PLAN

Arturo Molina

CSIM-ITESM*

9.1 Introduction

Nowadays, it is becoming more and more important to conceptualise businesses that are competitive to face the trends that are leading to increased levels of complexity, dynamism, and to uncertainty of the global environment. Traditional ways of designing businesses are more suited to evolutionary change, therefore the definition (or re-definition) of a business should take into account novel approaches from the start, i.e. the conception of the new or changed business. A business is the integration of people, processes and technologies with the aim of fulfilling some mission (usually the provision of a service or the production of some goods for a market). People who have entrepreneurship, leadership and the ability to collaborate, create an environment of trust and commitment that implements the envisaged concept. Therefore, the development of an enterprise concept is based on humans aiming to achieve their vision for an enterprise. Only people can recognise the customer's requirements, design products to satisfy them, and construct and run the factories that deliver those products. Only people can learn the lessons of experience and apply them in a manner that improves the future of a company (Miller and Berger , 2001). Hence, people play a key role in developing the enterprise concept; in fact, it is the people engaged in this activity who create enterprises. These enterprises will produce wealth for all stakeholders, people (customers, suppliers, employees, investors) and communities where the company operates and their products and services are used. It is therefore important to understand that the development of a business plan is about people interacting and collaborating in search of an enterprise vision.

* Centro de Sistemas Integrado de Manufactura - Instituto Tecnologico y de Estudios Superiores de Monterrey (Integrated Manufacturing Systems Centre - Technological and Superior Studies Institute of Monterrey), http://csim.mty.itesm.mx/

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

9.2 Strategy Schools Revised

It is imperative that any kind of business must be driven by a strategy. Mintzberg and Lampel (1999) have reviewed ten schools of strategy formulation. Strategy has a specific role in enterprise management. While the enterprise concept defines what the company intends to do (mission) and what the future will look like (vision, culture), it is not possible to create a detailed action plan from these statements about the future. The reason for this is simple: during the time period necessary to implement the future vision there are a large number of unknowns that make detailed planning impossible - hence feasible plans can only be created for the short term future. Strategy is the means to create coherence between the short term plans. When planning decisions need to be made, or risks need to be faced, strategy provides guidance regarding *how* and *why* to chose a particular course of action. Thus, strategy is not a set of steps to achieve the future objective; it is rather a distillation of principles that will be followed when plans are created.
In the methodology proposed in this chapter, contributions and ideas from different schools are incorporated in order to have a more complete approach to an enterprise concept definition. Contributions and ideas have been included from the following schools:

- Cognitive School, aiming to understand the process of strategic decision-making. Based on this concept the framework presented in this Chapter defines three major processes for enterprise concept development: creation of business concept, formulation of strategies, and the definition of an action plan.
- Design School, that perceive strategy formation as achieving the essential fit between internal strengths and weakness, external threats and opportunities. A SWOT[1] analysis is important to help in the definition of a feasible mission and vision.
- Entrepreneurial School, which considers the vision as the principal aspect of an enterprise concept. A unique vision of the future has to be defined and agreed on, with a leading figure and creative leader, who is the source of this mission and vision.
- Cultural School, defining the importance of culture as an enabler of strategy formulation.
- Positioning School, which provides an analytical approach to planning where strategy aims to define generic positions through formalised analyses of industry situations. The industry analysis based on Porter's five forces analysis is key in the formulation of strategies.
- Planning School, contributing with well structured planning process with different levels of decision making. The different levels allow decision makers to analyze various perspectives of strategy, such as corporate, business

[1] Strengths, Weaknesses, Opportunities and Threats.

and functional aspects. This idea is developed further to define competitive strategy, value chain strategy and production/service strategy.

- Power School, that has focused on strategy making based on power. Two approaches to power exist in a company:
  - Macro power, where the company uses its power over suppliers or customers, therefore a value chain strategy emerges of how a firm[2] will manage these relationships.
  - Micro power, which defines the company culture to elaborate principles and policies which guide the relationships among people.
- Learning School, which explores the model of strategy making as a learning process where learning about the resources, capabilities and competencies is an important part of the process and underpin strategies that continuously emerge. The use of the concept of Core Competence is a key in the development of an enterprise concept.
- Environmental School, which understands that companies and strategies have a life cycle, therefore it is important to face demands of- and on the environment to evolve, and for a viable strategy to exist.
- Configuration School, which views strategy as a transformation process for the company in order to undertake a revolutionary change. Strategy therefore is an important component of the perpetual transformation of the enterprise.

All these schools have been considered to define the framework for Enterprise concept definition described in the following sections.

9.3 Framework for Enterprise Concept Definition

The framework defined for the enterprise definition includes the following concepts (refer Fig. 9.1):

- External Drivers: all the issues that drive the environment and that will affect the decision maker in conceiving the business definition. This might include: Global Economy, Industry Context and Technology Evolution, Market Attractiveness, Customers and Suppliers Relations, and Government Influence.
- Internal Enablers: key aspects that might affect the realisation of the concept, which are internal to the company and have to be understood: Product/Service Life Cycle, Core Processes and Core Competencies (i.e. Human Capital, Technological Capital, Organizational Culture).
- Key processes: enterprise concept definition is a step-by-step process, therefore it is important to define its core constituent processes: business concept creation, strategy formulation and action plan definition.

[2] in the following, the terms 'firm', 'company' and 'enterprise' will be used interchangeably.

- Driving forces: enterprise is based on people, therefore major drivers in people's activities are considered, such as entrepreneurship, leadership and teamwork.

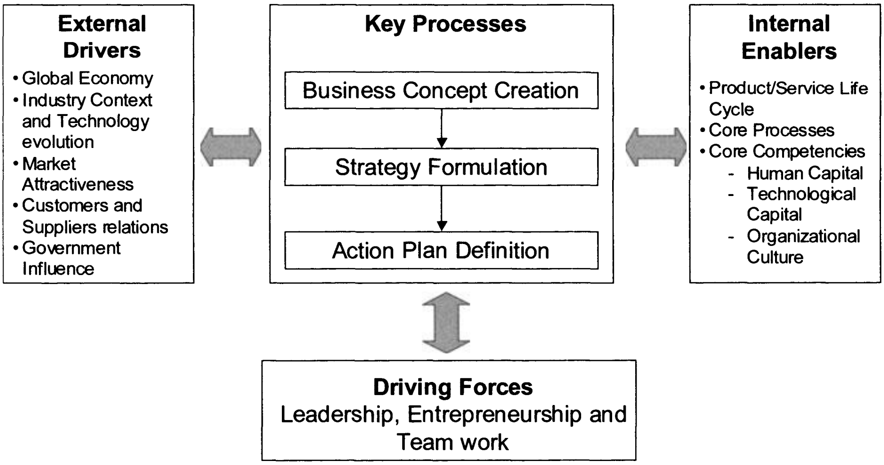


Fig. 9.1. Framework for enterprise concept definition.

9.3.1 External Drivers

Important drivers of the company decision making process are the external drivers, which influence the company. The following are important issues to be considered:

- Eight forces are shaping the global economy around the world and should be considered in the business definition. According to Sternberg (1999), these are:
  - Access, transformation and use of information, where people and organizations are becoming information processors;
  - The need to fulfil consumers' desires - business are to become customer-centred organizations;
  - Renaissance of capitalism as a unique mode of economy: companies are driven by profit;
  - Increasing internationalisation of commerce, the fall down of state walls;
  - Creation of worldwide corporations that are dominating national, regional and global economies, often overtaking government roles in national and international markets;

  - Emergence of a new generation of entrepreneurs focused on specialised competencies, creating networks of knowledge;
  - Rise of social movements penetrating business and government agendas related to human rights, ecology and sustainable development;
  - A threat imposed by the industrial society on clans, ethnicities and religions. Hence the need for understanding and tolerance.
- Burgelman et al. (1995) describes the following important aspects that will influence business decisions related to industry context and technology evolution:
  - Industry structure is guided by consumers, suppliers, potential competitors, product substitutes and competitors;
  - Markets are driven by technological innovations;
  - Complementary competencies are required to commercialise new technology, products or services;
  - Technological leadership is based on dominant designs;
  - Value added is determined by specific technologies;
  - De-facto standards in industry are emerging and should be considered;
  - There is a convergence of industries and it has an impact on society;
  - Technological change is interplaying with social systems and has influenced people's decisions.
- "All organizations are market-driven, whether they acknowledge it or not" (Moore , 1999). Therefore, market attractiveness is a must to pursue, if one has the vision of having a successful company in the market place. The attractiveness of a market is guided by profiles of customers, communities, regions and countries matched against the products, services and technologies that a company has to offer. In the definition of a business plan, a target market must be well specified, evaluated and its possible behaviour must be predicted.
- A new type of enterprise, the Extended Enterprise (Browne et al. , 1999), has emerged, which considers the relations with customers and suppliers as key for success. The company is no longer considered a four wall factory but a continuous process that encompasses suppliers and customers. In many cases, a company would want to respond to a market opportunity, but it did not have all the knowledge, physical capabilities, time, or financial resources required to produce the product. In these cases, the company will have to collaborate with others to complement its knowledge and capabilities.
- Government Influence: There are four major roles played by government that affects enterprise development (Osborne and Plastrik , 1998):
  - Policy maker;
  - Regulatory institution;
  - Service agency, and
  - Compliance organization.

All these four roles should nurture business creation if the right policies are defined (i.e. free market zones, economical agreements), rules and procedures to follow are well described (e.g. tax regulations), government services are effective and efficient; as well as all compliance functions are structured and customer-oriented (e.g. tax audits, occupational safety and health inspections). Therefore, it is important to be informed of policies, regulations, procedures and observe these during the enterprise concept development.

9.3.2 Internal Enablers

An enterprise should be able to deliver value added to a customer in the form of products or service. Therefore it is important to analyse the Product/Service Life Cycle. Different opportunities might arise if a thoughtful analysis of the life cycle of the products and services is considered (Levitt 1965). Depending on the product/service life cycle stage (introduction, growth, maturity or decline) the company must consider how this can affect the operation of the firm, the creation of new products/services or the development of new businesses. This analysis must be carried out in order to establish the firm's competitive position.

An enterprise employs *processes* to deliver products and services. A company must design its processes, comprising all activities required to meet customer requirements. These value added processes are also known as Core processes. Core processes are cross-functional processes that extend across the organization and directly result in the delivery of value to customers (Ostroff , 1999). Core processes must incorporate the best people, practices and technologies of a firm in order to achieve excellence in their execution. Well-defined and structured core processes are key enablers for a company to succeed. Therefore an important aspect of developing the enterprise concept is the definition of the Core Process required to offer unique value to the potential customers. Core processes are composed of a set of activities that are carried out by people working with technologies and supported by an organization *culture* (which includes values, policies, strategy, best practices and performance measures). All these elements can be configured in order to differentiate a company with unique Core Competencies. Core competencies are the result of collective learning of the organization (Prahalad and Hamel , 1990). They include people's knowledge and learning skills, owned technological resources, organizational capacity to organise, coordinate, integrate and deploy human and technological capital. An enterprise must be designed with a good understanding of the required Core Competencies that will enable a company to add value to the product(s) / service(s) and deliver it to customers.

Product/Service Life Cycle, Core Processes and Core Competencies are key concepts, which underpin the formulation of an enterprise based on internal strengths that must be aligned to the mission and vision of the business.

9.3.3 Key Processes in Developing an Enterprise Concept

The processes defined in the methodology to develop an Enterprise Concept are (Fig. 9.2):

- Creation of Business concept: is the process that includes all the activities required to define the mission and vision, and which aligns these with organizational culture. Cultural alignment is bi-directional - present culture determines the feasibility of the mission and vision, while a new mission and vision can influence company culture.
- Formulation of Strategies: is the process focused on determining the strategies of the business being created. Three types of strategies have been defined to be of relevance: competitive strategy, value chain strategy and production/service strategy.
- Definition of an Action Plan: is the process that defines the core processes and competencies of the envisioned enterprise. It also creates the specific action plan required to establish the company (or to change the existing one) and formulates the final business plan.

9.3.4 Driving Forces: Entrepreneurship, Leadership and Team work

The strength of an enterprise is based on its people, is reflected in its identity and in its ability to leverage that identity with the purpose of creating products and services that provide value to customers. The value production process necessitates co-operation, and co-operation requires compromise (mutual agreement) between people. This is a key factor of a successful enterprise. Entrepreneurship, Leadership and Teamwork are driving forces that are needed to achieve co-operation and mutual agreement.
Entrepreneurship is the driving force in the creation of an enterprise. It has to do with aspirations, dreams, vision and commitment to an idea or ideal. Entrepreneurs must face three major questions:

- Are my goals well defined ?
- Do I have the right strategy ?
- Can I execute the strategy ?

An entrepreneur must clarify current business goals and align them with personal aspirations. The business must be defined so as to be sustainable, and there is a need to evaluate how much risk is acceptable for implementing the idea. Regarding the second question, the right strategy provides a clear direction for the enterprise. The entrepreneur's vision defines the strategy for the new company and must define where the company is going to be in the future - regardless of where it is presently. The third question is separate (albeit related): it must be determined whether there is an ability to access resources,

to build up organizational capabilities, and to clearly define personal roles of the people involved in the venture (Bhide , 1999).
In case of an existing business, once the company is established, an entrepreneurial management is considered key for the success of the operation of the business. This type of management requires the following principles (Drucker , 1985):

- Change is an opportunity, not a threat. Innovation is a must;
- Improve performance through systematic measurement and learning;
- Use a structured and transparent system for compensation, incentives, and rewards.

Leadership is required to design, create and run a successful enterprise. Leadership must deal with change in the organization. Leadership must set a direction based on a vision, should communicate and align people to it, and must motivate and inspire people to pursue that vision (Kotter , 1998). Leadership must be participative and inspire teamwork. Leaders involved in the definition of an enterprise concept must have wisdom that combines knowledge and judgment; have the integrity to be direct and honest, thereby inspiring trust. Finally, they should have the courage to tackle uncertainty and, if required, being able to share responsibilities and power (Zand , 1997).
Chapter 5 is separately dealing with leadership, and carries two important messages.

- Leading and directing are antagonistic - a leader pulls people in a direction - inviting them to follow, while a 'director' applies push - forcing people to go into a given direction. Psychologically, people respond much better to the first kind of treatment;
- If pull does not work (people do not respond to the new idea), then changing the behaviour of people through the introduction of new processes, for example, is easier then convincing them through theoretical arguments. If a leader needs to institute change, then it is a better strategy to implement new processes and let people realise its benefits (and accept it through personal experience), then it is to force them to accept arguments to which they can not relate due to lack of experience with the new situation.

Teamwork requires integrating people with different skills, preferences, experiences and knowledge. Visionaries, managers and operators must compose a team in order to build up an enterprise. Visionaries are required to explore new ideas and push the team effort towards new horizons. Managers are the business builders who can plan, control and articulate actions to achieve specific goals and objectives. Finally, operators are doers, highly skilful people with deep knowledge of the business who can carry out all the necessary actions to attain the enterprise vision (Baghai et al. , 1999).
The ability to create a competitive organization depends on the entrepreneurship, leadership and teamwork of the people involved in this process.

9.4 Methodology for Enterprise Concept Definition

- The methodology defined in this Chapter is based on three key processes (refer Fig. 9.2):
- Creation of Business concept,
- Formulation of Strategies,
- Definition of an Action Plan.

These three processes allow decision makers to follow a set of activities with specific results. Figure 9.2 shows the major outcomes of each activity. The creation of the business concepts will result in the definition of the mission, vision and intended cultural attributes (values and policies) for the company. The strategy formulation process aims to define three types of strategy that will drive the action plan definition. The strategies to be formulated are competitive strategy, value chain strategy and production / service strategy. All strategies must derive into an *action* plan and its associated *business* plan. Each of these strategies may be further subdivided, e.g. competitive strategy may include both product and resource strategy, and resource strategy may be further subdivided into human resource, materials, information and knowledge management strategy, and asset strategy. In order to avoid being lost in details, there should be a small number of leading strategies. The rest may be relegated to planning, so as to establish what other lower level strategies are necessary to make the main strategy feasible. In this manner, a company can develop a hierarchy of strategies without losing sight of the main strategic aims.
An action plan for a company begins with the definition of the *core processes* required to achieve the desired behaviour of a company in order to deliver products of services. These core processes uses resources and capacities (Human and Technological Resources) to create unique core competencies, which will enable a company to achieve the desired strategy. Finally, a business plan must be created in order to describe in detail the complete concept of the enterprise to be created.

9.5 Creation of the Business Concept: Mission, Vision and Intended Set of Cultural Attributes

Elements that are relevant in the creation of a business concept are the definition of the business mission and the articulation of a vision, as well as the establishing of the desired culture that will be guiding decisions in the future materialization of the enterprise. The creation of the business concept also requires analysing the external drivers and internal enablers of the company, in order to have a complete view of all the aspects that will influence the firm. A SWOT analysis is a recommended tool to be used in preparation for the conceptualisation of the Mission, Vision and Culture (refer Fig. 9.3). The

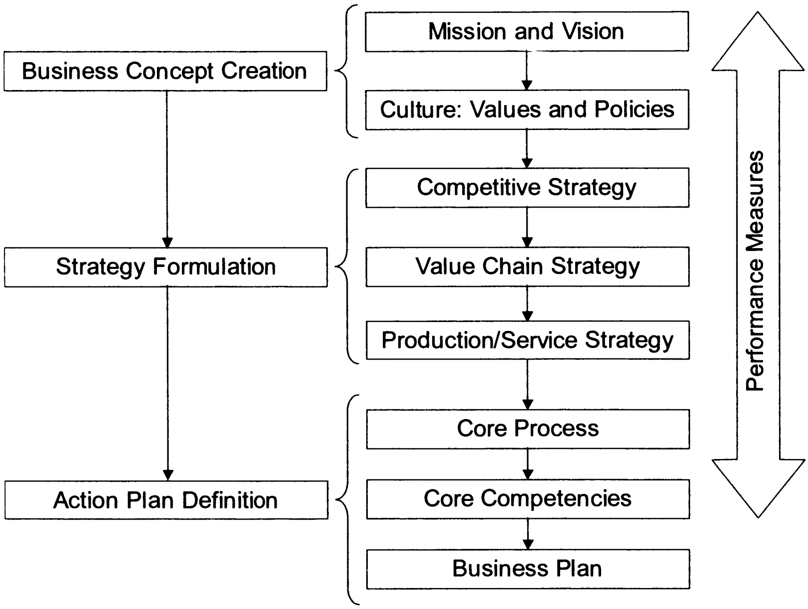


Fig. 9.2. Framework for enterprise concept definition

SWOT framework clarifies the gap between where the firm is and where it should be, based on an internal and external analysis of critical issues. Importantly, a SWOT analysis can only be carried out if the desired future situation is known, because this analysis measures the company's position relative to its intended position. The result provides decision makers with a wide-ranging understanding of the organization's strategic concerns.

9.5.1 Mission: the Reason for Being

Mission statements can take different content, format and specification. David (1995) describes nine characteristics that must be included in a mission statement:

- Customers: who are the firm's customers ?
- Product or services: what are the firm's major products or services?
- Markets: Where and with how does the firm compete ?
- Technology: Is technology a primary concern of the firm?
- Concern for survival, growth and profitability: Is the firm committed to economic objectives?
- Philosophy: what are the beliefs, values, aspirations and philosophical priorities of the firm?
- Self-concept: What is the firm's distinctive competence or major competitive advantage?

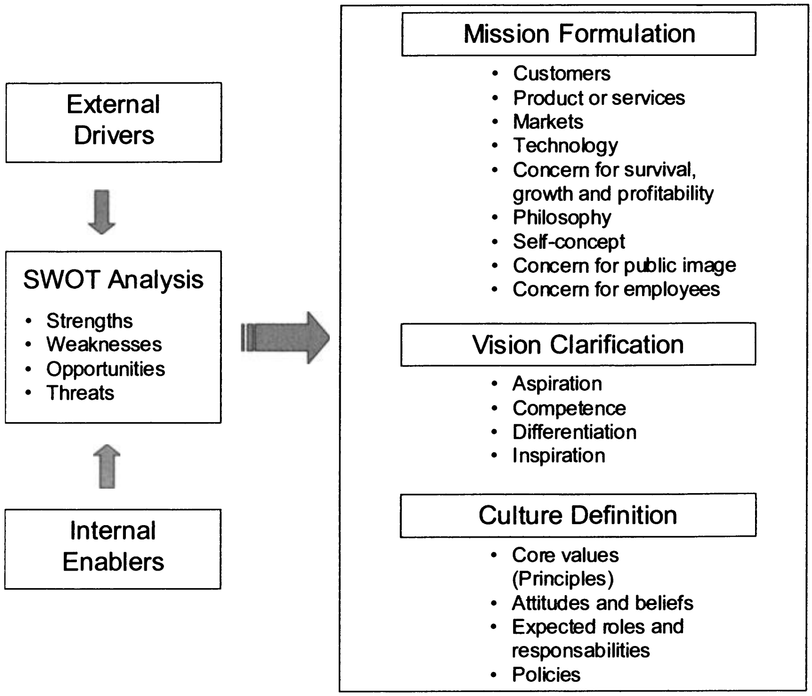


Fig. 9.3. Business Concept Creation

- Concern for public image: Is the firm's responsive to social-, community- and environmental concerns?
- Concern for employees: Are employees considered to be a valuable asset of the firm?

Rea and Kerzner (1997) recommend that the formulation of a mission statement will be better written *after* a thorough analysis of the strategic issues, competitive advantages, position of products/services and customers' perceptions have been performed. By doing so, the mission statement will include the key aspects of what the company really is, and can demonstrate how or why the company is unique and distinctive. Consequently, a better time to define a mission statement is usually after a SWOT analysis has been undertaken. The question that might arise is who should participate in the definition of a mission statement? Creating a well-defined mission for the company is not just one person's responsibility. It is the responsibility of everyone in the organization. People involved in the creation of the company, have to contribute to the mission, but it is important to create a team effort to focus on developing a mission statement that represents what the company is and what the company believes in. Important guidelines for the formulation are: keep it simple, short, make it timeless, and make it personal. The mission will have

to motivate and give a real meaning to the work that people accomplish in the company.
The definition of a mission has been a popular management tool. A survey carried out by Rodin and Hartman (1999) showed that 90% of US companies have defined a mission. However, less than half of the executives interviewed believed that their companies' mission statements were significant. Moreover, fewer felt that their mission was being actively pursued. Therefore, once defined, it is important to *communicate* the mission to all the stakeholders in the company, even to the customers. Mission Statements should be visible to all and practiced daily. A common practice has been to display it on every possible item of the company, for example business cards, letters, walls, etc. Remember that people who believe in the importance of the company's mission, will place the company's interests on a par with their own individual interests when effort and performance requires it. However, there is another reason why many mission statements are not as relevant as they should be, and it is connected with the mission statement being formulated on a too generic level, or including irrelevant aspects rather then referring to concrete markets, customers, services or products. Such mission statements are thus not providing guidance for decision making and are not very useful.

9.5.2 To be or not to be: Dimensions of Vision

The future for any company begins with a *vision*. A vision must answer questions like what the company will be like in the future, where the company is going, and what is the reason for the continued development of the company. It is important to have an inspiration to define a vision: this inspiration may come from understanding the company's stakeholders, the community in which the company exists, the competition, the direction of technology, social development, etc. There is no 'standard' or formula for creating a vision. There is uniqueness in the vision that really comes from the entrepreneur's ideal about the enterprise that is to be created. Successful visions feature at least one of four dimensions (Belasco and Stead , 1999):

- Aspiration: The company aims to accomplish certain objectives that create success for the stakeholders, including customers, suppliers, employees and investors;
- Competence: The company bases its vision on certain core capabilities that has a strategic intent (skills, knowledge and technologies);
- Differentiation: The company attempts to be unique and distinctive;
- Inspiration: The company endeavours to create wealth for the community where its services and products are used.

Collins and Porras (1997) have identified the following elements that could be included in a vision statement:

- Core ideology characterizes a company and this character will remain consistent through time;

- Core values are the set of timeless guiding principles that define the company's identity and gives confidence to those inside the organization;
- Core purpose, or the organization's reason for being. It captures the soul of the organization;
- Envisioned future to create processes for constantly renewing the company's business concept.

What a vision does for a company can be summarized in the following (Hoover , 2001):

- A vision can bring people together aiming at a shared purpose;
- A vision can be a continual motivator and force of inspiration;
- A vision is an anchor in hard times of change;
- A vision builds a sense of community.

In the words of Collin and Porras (1994) "To pursue the vision means to create organizational and strategic alignment to preserve the core ideology and stimulate progress toward the envisioned future." Therefore it is important not only to define a vision for the company but also to translate it to reality, through the definition of strategies and the formulation of actions organised in a business plan.
Finally, a note: Chapter 12 elaborates on the design of the management system of enterprises. When designing the management system, the strategic level management tasks elaborated here need to be designed into the system of management, including future re-formulation of the enterprise concept, and periodic (or event driven) re-consideration of all strategic components of the business concept - mission, vision, strategies and action plans. Especially in small and medium sized enterprises, these tasks are often not pursued systematically. As a result, companies may be caught out in face of changes in markets, products, technologies, availability of strategic information and of knowledgeable human resource, or financial market changes. Since small and medium sized enterprises do not always have the resources or the knowledge to complete these tasks alone, there is an important role of industry associations and government agencies that can provide input to the company's strategic decision making.

9.5.3 Culture: Driving Behavior

The ideas embraced by the mission and vision must permeate the behaviour of many people in order to move from a mission and vision to concrete results. Results emerge only if and when people in the company do act. For this to happen, a *culture* must be created in the organization. Company culture has become a popular summary term referring to the values, attitudes, beliefs, policies and customs widely held by people who work in a company. Culture includes:

- Values defined by things such as achievement, community, excellence, fun, integrity, innovation, safety, teamwork;
- Attitudes and beliefs of what to expect from the company, managers and employees;
- Policies, which define rewards and penalties within the company;
- Customs regarding how informal information flows, what is being said, and how decisions are made.

Every company has a culture, whether it is defined formally or informally. Usually, manuals and policy statements describe the formal part of the culture of an organization. Culture is the identity of the organization, it supports change in an organization, and allows people to work together in an environment defined by laws. Ackerman (2000) has defined the Laws of Identity that form a set of value-identifying guidelines for any organization to create its own culture:

- The Law of Individuality asserts that every organization is unique and has differentiated value-creating characteristics. Therefore they define a clear mission.
- The Law of Will drives people and companies to be the best at whatever they do. Therefore declare a shared vision and become a true leader.
- The Law of Possibility defines the potential for timeless value creation of a company. Therefore constantly be innovative and entrepreneur.
- The Law of Being reveals that organizations are comprised of unique individuals with their own thoughts, ideals and beliefs. Therefore managers must be responsive to employees' needs and show their appreciation.
- The Law of Comprehension expresses that individual capacities of the organization are only as valuable as the perceived value of the whole of that organization. Therefore they foster teamwork.
- The Law of Relationship supplies the basis to create value for all stakeholders (customers, employees, investors). Therefore build long terms relations based on common values and wealth sharing.
- The Law of the Cycle notes that companies are part of a wealth creation cycle and must endure their relations with the community they work for. Therefore make work-life balance a core value in the organization.
- The Law of Constancy declares that identity is fixed, transcending time and place, while its manifestations are constantly changing. Therefore live by genuinely felt values and operating principles.

The starting point in the developing of the Enterprise Concept is the answer to the question of why the business exists. The mission and the vision underlie the concept of the business purpose. Therefore, they are the basis for the creation of every new enterprise. The organizational culture is the driver of the behaviour of a company in order to fulfil the mission and vision. For that reason, culture is the key for the successful development of an enterprise.

9.6 Development of Strategy: A Decision-making Process

Strategy conception is a decision-making process that enables a company to achieve its mission and vision. There are two stages in this decision-making process: strategy analysis and strategy formulation. Grant (2002) mentions three types of analysis to be considered to formulate strategies (refer Section 9.6.1):

- Analysis of industry and competition
- Analysis of the firm
- Analysis of competitive advantage

The analysis of industry and competition examines the external drivers (Global Economy, Industry Context and Technology Evolution, Market Attractiveness, Government Influence) that are important to the firm, and evaluates how they are likely to impact the company's relations with customers and suppliers.

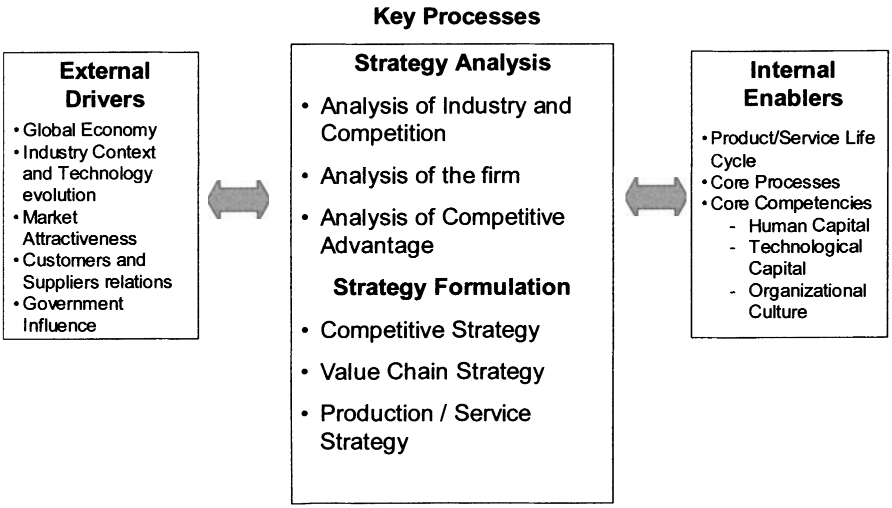


Fig. 9.4. Strategy decision process: analysis and formulation

The analysis of the firm concentrates its attention on the internal enablers of the company (product / service life cycle, core process and core competencies) and assesses how they can be aligned to support or to achieve competitive advantage. Note that in case of developing a new company this analysis is postponed: although this analysis is part of the development of the enterprise concept, the analysis is carried out only after an action plan has been defined.

(The reason being that there is no company yet, therefore there are no restrictions regarding the internal enablers, thus the enterprise can be conceptualised without internal constraints.)
The third type of analysis explores how the company can sustain a competitive position in the marketplace using different strategic approaches, such as innovation, cost leadership or differentiation.
Once the analysis of competitive advantage has been performed, the formulation of the strategy for the company must be able to answer the following key questions:

- Which industry should the company be in?
- How should a company compete in the selected industry?
- How should core process and core competencies be aligned in order to become competitive?

Traditionally, these levels of strategy have been named corporate strategy, business strategy and functional strategy respectively.
Three different types of strategies have been defined in this Chapter, in order to support the creation of the enterprise concept:

- Competitive strategy defines in which industry the company will compete and how a competitive advantage can be achieved.
- Value Chain strategy defines how a company will organise its value chain (i.e. relation with suppliers and customers) in order to compete in a selected industry.
- Production / Service strategy defines how the company will organise its core processes and competencies and align them to accomplish the selected competitive and value chain strategy.

It is important to reinforce that the decision making process for strategy making requires both analysis and synthesis.

9.6.1 Analysis for Strategy Formulation

The analysis of industry and competition and the analysis of competitive advantage are reviewed in this section to explain to the reader how these analyses can be performed using various analytical tools.

9.6.1.1 Analysis of Industry and Competition

The external drivers (Global Economy, Industry Context and Technology Evolution, Market Attractiveness) influence the company's decisions and performance. Because there is too much information to analyze, it is important to distinguish those external drivers that are vital for the company. Porter's 'Five Forces of Competition' framework has been used to analyze those drivers

(Porter , 1979). Porter's five forces include suppliers, buyers, substitutes, potential entrants and industry competitors. The relations between these five concepts and the firm's position can be expressed in terms of (1) bargaining power with suppliers and buyers, (2) threats posed by substitute products and new entrants, and (3) rivalry with existing firms. This analysis allows a company to make decisions by answering the following questions:

- What are the possibilities to enter or sustain the presence in a specific industry sector, based on the level of concentration of companies, diversity of competitors and available information on profit-attractiveness?
- How difficult is for a company to enter-, or sustain its presence in an industry sector in terms of capital requirements, possibilities of economies of scale, cost advantages, access to channels of distribution, government and legal barriers, possible retaliation by established firms?
- How difficult will is it be to exit an industry sector in case of future unsuccessful business? (E.g. because of commitments made to stakeholders).
- What is the potential for product/service differentiation?
- What will be the value offered to customers?
- How will it be possible to establish good relations with suppliers?

These variables determine the target industry's potential for making profit, and useful data can be obtained by analysing trends in demand and competition. The identification of key success factors is vital for the success of an enterprise by defining what customers want and what does the firm need to do to survive competition.

9.6.1.2 Analysis of Competitive Advantage

"Competitive advantage is the ability of the firm to outperform its rivals on the primary performance goal – profitability" (Grant , 2002). Competitive advantage emerges from:

- External sources: changing / influencing customer demand, prices or product / service technology.
- Internal sources: capacity of the firm to innovate in products/services and processes.

Changes in external sources are happening every moment in today's dynamic markets. Monitoring these changes is a necessity for a firm and can be carried out using methods of competitive intelligence (Shaker and Gembicki , 1998). Questions regarding changes must address the following issues:

- How is customer demand changing?
- What is the profile of the firm's customers, markets or target industry?
- Is it important to compete based on costs or product differentiation?
- Do customers demand innovative products? New technologies?

- Are there any significant changes in the value adding process?

The firm's strategy will define how a firm intends to sustain its competitive advantage by defining and understanding the firm's competitive edge, its role in the value chain and through its production/service delivery strategies. The role of the above analysis is to help investigate the feasibility of these strategies, and – in case of multiple strategic choices – to help management make an informed decision on the strategies to be adopted.

9.6.2 Formulation of Strategies

The order of strategy formulation and strategic analysis cannot be prescribed as a sequential process. Under some circumstances, management needs a preliminary analysis so as to find possible strategies. However, a preferred order would be for management to formulate an initial set of alternative strategies, followed by a deeper analysis of their feasibility. Strong managers who live in the industry, through their experience and immersion in the affairs of the company as well as its industrial context are often capable of identifying initial strategies, which would then have to be investigated as for their implementability. It is best to think of this process as a parallel iterative one, where the synthesis and analysis are performed in parallel, and refine their results until agreement is reached on the main points. Whether the analysis preceded the initial strategy formulation or not, in order to arrive at a strategy the selected strategies need to be formulated in a final form and must be communicated.
The methodology for enterprise concept development presented in this Chapter defines three types of strategies: competitive strategy, value chain strategy and production / services strategy (Fig. 9.4). The following subsections present guidelines for the formulation of these strategies.

9.6.2.1 Definition of Competitive Strategy

Hope and Hope (1997) propose that for a company to achieve competitive advantage, at least one of the three following properties are necessary: operational excellence, product leadership or customer intimacy. These propositions are related to Porter (1980), and offer three generic choices as candidates for the company's competitive strategy: cost leadership, differentiation and focus. Cost leadership is a strategy driven by offering low prices. This strategy is often selected when the analysis of industry and competitive advantage verifies that a set of customers or markets are very interested in buying products and services on the basis of low prices and convenience. Therefore the focus of the company is to achieve operational excellence in order to be able to deliver products / services at higher efficiency so as to drive costs down. A low-cost producer must find and exploit all sources of cost advantage. Cost advantages

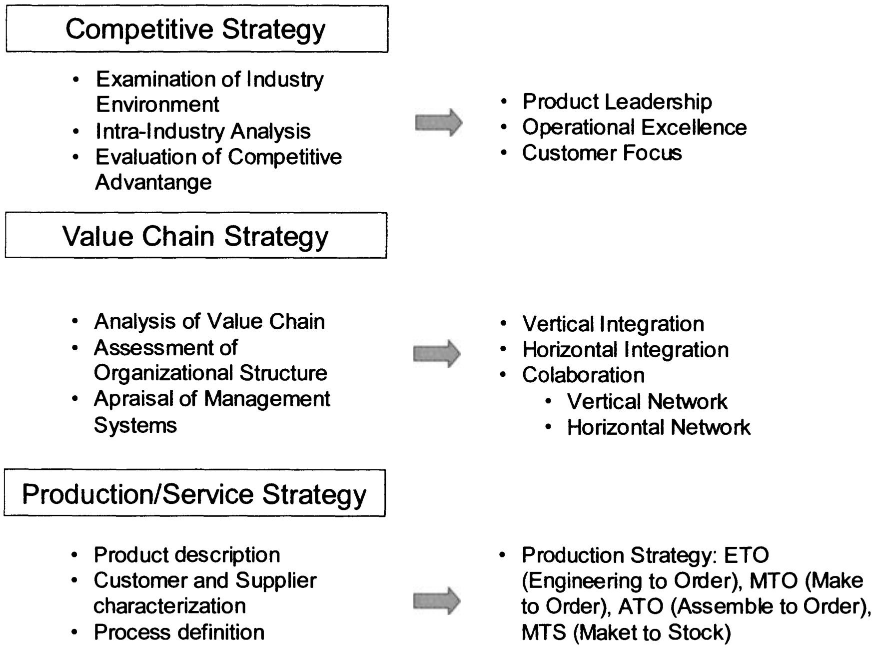


Fig. 9.5. Strategy Formulation process

often require relatively high market share or advantages such as access to essential raw materials, cheap labour, faster distribution channels and proximity to customers. Process innovation is important to sustain low-cost positions. Low-cost producers typically sell a standard, or no-frills, product and place considerable emphasis on economies of scale or absolute cost advantages from all sources. Examples of international companies using this strategy are: GE (six sigma), McDonalds (local supplier network development) and Chrysler (extended enterprise model).
The second strategy is to differentiate the product, creating product leadership and an image to be perceived in the industry as unique. There can be a wide range of unique features such as brand image (Nike), innovation (3M), technology (Sony), or distribution efficiency or reach (Federal Express). Differentiation can create brand loyalty and lower sensitivity to price. It can increase profit margins and reduce the need for controlling costs. When a company decides to pursue leadership in products and services it should be aware that this might represent a limited market share, since it often requires a perception of exclusivity. However this strategy implies the ownership of innovation capabilities to constantly develop new products and services. Companies pursuing this strategy are: Microsoft, 3M, Sony, Intel, Merck, and Nike, among others.
'Focus strategy' or 'Customer intimacy' is the third generic strategy. It focuses on a specific market segment and selects a set of customers interested in

getting exactly what they want. This strategy assumes that such companies can more effectively and efficiently serve a particular customer or market then its competitors. In essence, the firm provides a unique product / service to its target markets at a reasonable cost. The company concentrates on specific customers and satisfies their special needs. A trade-off between profits and sales volume is required when a company aims to be specialised. Companies following this strategy are: Levi-Strauss (customised jeans), USAA (loyalty management), Honda (high quality services) and Staples (on demand supply of office goods).
The development of the enterprise concept requires the definition of a 'value proposition', underpinned by core processes and competencies. In order to deliver this value to customers the following guide may be used:

1. Define what is important to customers.
2. Describe how profit can be made from offering customers the value added.
3. Decide what strategy matters most to customers (costs, differentiation, focus).
4. Lists all the choices available now and in the future to deliver value added based on the selected strategy.
5. Choose the ones that can be accomplished with available (or feasible / accessible) knowledge, capabilities and resources.

Following this simple guide allows the company to define a good competitive strategy as part of the enterprise concept.

9.6.2.2 Definition of a Value Chain Strategy

Once the competitive strategy of the company is understood, it is possible to translate it into a set of decisions about how the organization can deliver its 'value proposition' to the customer. Value Chain strategy is about making decisions on how a company will establish an organizational model that will best exploit its potentials and opportunities to build an effective and efficient value chain.
Companies cannot deliver added value alone; this can only be achieved through participating in a value chain. Hence, they must consider active participation in the establishment of a value chain that allows the company to optimise its resources and capabilities in order to attain competitive advantage. The Value Chain strategy analyses the possibilities for the creation of a strong value chain and assesses the organizational structure and management system of the company to decide on a viable strategy or set of strategies to pursue.
Different directions can be considered and adopted as a value chain strategy:

- Vertical Integration: this occurs when a firm moves forwards or backwards in the value-adding chain through acquisition or merger with other companies;

- Horizontal Integration: this refers to a strategic direction intending to grow through the acquisition of competing firms in order to have access to complementary process and competencies;
- Collaboration: a new trend which is based on the creation of a network of associated companies committed to act jointly to offer value, and is believed to be particularly suited for dynamic markets.

Vertical integration has been a common strategy for companies trying to gain control over the raw material supply, or over distribution channels. Backward integration usually lowers costs; therefore it supports cost related competitive strategies. On the other hand, forward integration has been considered a focus strategy to allow the firm flexibility in differentiation by gaining control of market processes and direct access to customers.

Horizontal integration focuses on accessing new processes or competencies to increase the possibilities of product innovation or technological leadership. By acquiring competing firms a company immediately gains access to new technologies or knowledge that can be incorporated into its line of products or services. Also this strategy allows a firm to have access to the acquired competitors' markets.

Collaboration based on dynamic industry networks is a relatively new strategy for creating value-added business. The main benefits of creating and participating in industry networks are described in terms of their capacity for apprenticeship and innovation and an improved ability to manage uncertainties. (refer Table 9.1).

There are two types of networks with well-defined characteristics: *vertical* networks and *horizontal* networks. Vertical networks are based on co-operation between enterprises that belong to different and consecutive positions in the supply chain, where the main objective is to achieve competitive advantages that could not be achieved by any of the individual participants. A typical example of a vertical network is based on strategic and stable relations established between one or several large enterprises and their small and medium sized suppliers. As opposed to this, a *horizontal* network is a compound of similar-sized enterprises that produce the same type of goods, and decide to unite for commercialisation, raw materials supply, or invest and provide themselves with common services. Horizontal networks can either be formed by enterprises that produce a single product, where each of the enterprises is specialised in one component of the product, or cater for different geographic regions (typical of the service industry). Generally these networks are groups of small and medium sized companies from the same industry sector, looking at economies of scale, and at increasing their negotiation power.

Table 9.1. Main benefits of industry networks (PNUD , 2000)[3]

Capability	Network Results	Benefits
Learning and innovation capacity	Increase capabilities for capturing, selecting and effectively using information. New product development. Flexibility. Improved quality standards.	Exchange of experience and information. Extension of contacts through the network. Improved ability to develop complementing specialised capabilities and better productivity.
Strategic management capacity	Efficacy: improved capability to deal with uncertainty.	Complement / replace individual functions with collective functions through: • Agile market behaviour. • Information routing by systematic information exchange. • Reputation / quality certification. • Acquisition, joint venture.
Scale economies and commercial negotiation power	Incorporation of high cost technologies. Raw material costs reduction. Stable subcontracting relations between small and large enterprises.	Group investments Supplier negotiations. Group commercialisation. Common services. Negotiation with financial entities.

The creation of industry networks has allowed the conceptualisation of a new type of company called the Virtual Enterprise (Molina et al. , 1998). Chapter 7 covers the issue of enterprise network creation and virtual enterprises in more detail.
The definition of a value chain strategy defines how the enterprise can find a suitable organizational model to face the challenges of competition.

9.6.2.3 Production/Service Strategy Definition

A production/services strategy is based on the following factors:

- Product description : defines criteria required for a company to qualify or to win an order in a specific market.
- Customers and Suppliers characterisation: defines customer's expectations and requirements imposed on suppliers.
- Process definition: specifies performance measures required in the execution of the activities in the process.

[3] with some extension to the original reference.

All these factors are defined by order-qualification and order-winning criteria (Hill , 1989). The criteria are: price, volume, quality, lead-time, delivery speed and reliability, flexibility, product innovation and design, and life cycle status. Based on all these performance measures the following production strategies can be defined (Rehg , 1994):

- Engineering to Order (ETO) defines a company that has a product that is either in the first stage of the life-cycle curve, or a complex product with a unique design (OKP[4]) produced in single-digit quantities. Examples of ETO include construction industry products (bridges, chemical-plants) and large products with special options that are stationary during production (commercial passenger aircraft, ships, high-voltage switchgear, steam turbines). Due to the nature of the product, the customer is willing to accept long lead-times because engineering design is part of the process.
- Make to Order (MTO) assumes that all the engineering and design are complete and the production process is proven. Manufacturers use this strategy when demand is unpredictable and when the customer lead-time permits the production process to start on the receipt of an order. New residential homes are examples of this production strategy. Some computer companies make personal computers to customer specifications, so they follow MTO specifications.
- Assemble to Order (ATO) is adopted when the customer lead time is less than manufacturing lead time. The automotive and electronic industry are following this strategy. This strategy is used when the option mix for the products can be forecast statistically. In addition, the subassemblies and parts for the final product are carried in a finished components inventory, so the final assembly schedule is determined by the customer order.
- Make to Stock (MTS) is used for two reasons: (1) the customer lead time is less than the manufacturing lead time, (2) the product has a set of configuration and few options so that the demand can be forecast accurately. If positive inventory levels (the store shelf is never empty) for a product is an order-winning criterion, this strategy is used. This option is often related to the maturity stage of the life cycle and usually occurs at maximum production volume.

This strategy defines the criteria that must be satisfied by the company in order to be able to compete in the selected markets and industries.

9.7 Definition of an Action Plan

The final stage in the developing an Enterprise Concept is the definition of an action plan to create the enterprise or to transform the existing enterprise into its future desired state. Important activities to consider are (refer Fig. 9.5):

[4] One Kind of Product

- Core Process Selection and Design:
  - Design core processes;
  - Determine performance metrics;
  - Associate human and technological capital;
  - Design organizational structure.
- Core Competencies Selection and Development
  - Identify resources and capabilities;
  - Evaluation of resources and capabilities;
  - Selection of capabilities and competencies;
  - Define action plan to develop competencies.
- Business Plan preparation.
  - Define action plan;
  - Write business plan.

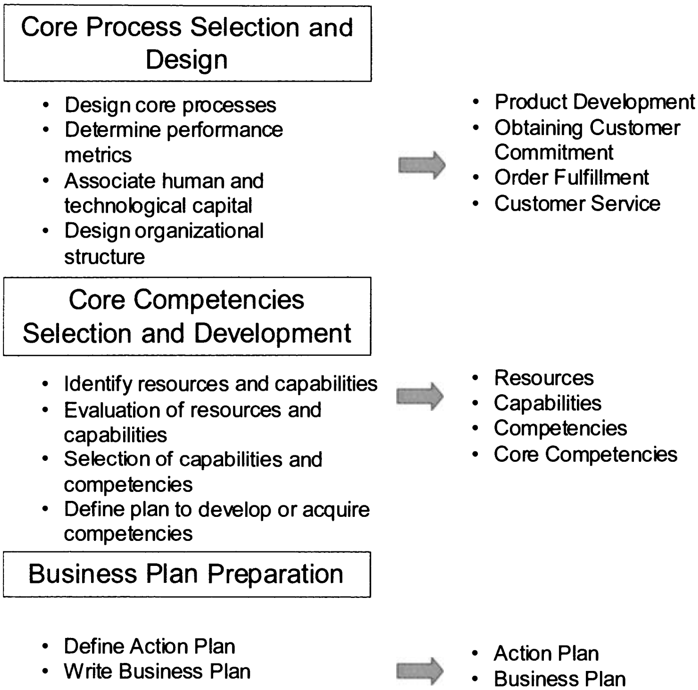


Fig. 9.6. Action Plan definition

9.7.1 Core Process Selection / Definition

The internal enablers (product/service life cycle, core processes and core competencies) are evaluated to select and design the core processes. Three cases can be identified:

- When a company is already in operation, the firm must identify in what stage of the life cycle their products / services are, in order to decide what business (products / services) can be pursued or which ones must be discontinued. Then, the evaluation of core processes and competencies can be carried out in order to align them to exploit the selected businesses;
- When new products or services are being conceived, the firm must evaluate how the existing core process and competencies can be used to deliver this new service or product. In the case that the company does not have the required core process and competencies, the best strategy to acquire them can be defined;
- When a new enterprise is being created, the product / service life cycle analysis of different markets allows entrepreneurs to assess or identify new business opportunity. Then, core processes and competencies required to exploit this business opportunity can be selected to implement and operate the enterprise.

The analysis of the firm begins with the evaluation of the product/service life cycle. Levitt (1965) defined four major stages in a product's life: introduction, growth, maturity and decline. The evaluation of a company's product/service life cycle helps to identify business opportunities. Some examples that can be drawn from this analysis are: when a product / service is about to reach the declining stage in the actual market, there is the possibility to reposition this product / service in a new market; when products are in maturity stage, a company can aim to be a low cost supplier for companies which are outsourcing their products; at growth stage products/services performance must be improved so as to defeat competition, or by constantly innovating and introducing products more rapidly than competitors. All these business prospects can be pursued by a company, provided the right strategies are defined and core processes and competencies are aligned to fit the strategies. In order to achieve the best strategic fit, it is important to analyse the core processes and competencies of the firm.
The development of core processes' models in an enterprise requires the alignment with the selected competitive strategy (Product Innovation, Operational Excellence, Mass Customisation), product / service strategy (ETO, MTO, ATO, MTS) and performance measures related to range of products, processes, sales, costs and/or production volumes.
Core processes must be designed if a new enterprise is being defined. A good guideline for defining what core processes should exist in the new organization is to use the model of the Extended Enterprise. The reference model used in this Chapter is based on the Extended Enterprise concept (Browne et al. ,

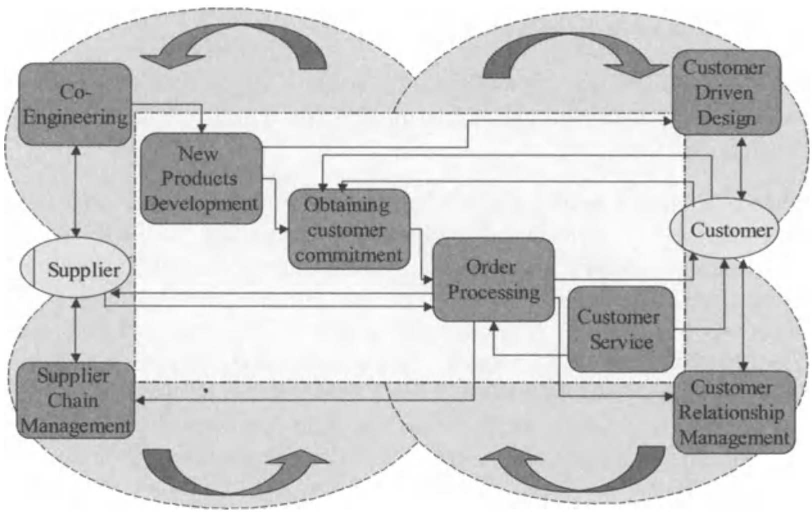


Fig. 9.7. Extended Enterprise Reference Model for Core Processes

1999) and the ENAPS Reference Models (ENAPS , 1999). It comprises eight business processes to describe a generic structure of an ideal intra- and inter-integrated extended enterprise (refer Section 9.7.2):

- Customer Driven Design. Includes all activities concerned with product design and engineering based on customer specifications;
- Co-Engineering. Encompasses all activities related to the management of supplier capabilities specifically related to product design and engineering;
- Supplier Relationship Management. Management, evaluation and monitoring of supplier capabilities related to production, logistics, and service delivery;
- Customer Relationship Management. Compile, analyze and share among enterprise stakeholders all information regarding customer needs and expectations;
- New Product Development. Includes all activities for creating new products and/or services and related manufacturing and logistic processes;
- Obtaining Customer Commitment. This is the process that brings the products or services to the customers. The process starts with market analysis and the identification of client needs and ends with the confirmation of an order;
- Order Fulfilment. Encompasses all activities concerned with processing orders coming from customers. The process starts with the order release from sales and ends with the product or services delivery;
- Customer Service. Focused on maintaining customer satisfaction after purchase is completed. This process deals with all activities from the delivery of a product or service to the point when it is no longer in use (end of life).

This general reference model can be a guide for the selection of core processes for a particular enterprise, and a set of performance measures can be selected in order to measure the outcomes of these core processes. A strong relation exist between the results analysing the life of the product or service and the definition of what the core process are (or should be) for a company. If a company identifies that a business opportunity exists to be a low cost supplier, then core process related to customer relationship management, obtaining customer commitment, order fulfilment and supplier relationship management must be considered in order to design the enterprise.
On the other hand, when a company is already operating, the following questions may help the company identify its necessary core processes (Ostroff , 1999):

- Does the process significantly impact the delivery of value to the customer?
- Does the process include activities, information and material flows that extend across functional, business, geographic and any other boundaries?
- Does the processes account for a major portion of the company's costs or revenues?
- Does the process have measurable outcomes?

If the answer is positive to all these questions then it is more likely that the process will be a core process. Identification and evaluation of core process delivers useful information because it helps the company redefine or re-define its value proposition.
Note that Core Process Selection as described in this Chapter is not the same as process engineering - on this level it is merely necessary to identify which major tasks need to be performed. These tasks are performed by the 'core processes'. The available options for the actual detailed specification and design of these processes are separately described in Chapter 15.
One way to ensure that Core Processes perform according to expectations is to define and use performance indicators (PI). Performance indicators are related to cost, time, quality and flexibility. Benchmarking is a very powerful tool for comparison and deciding if a core process of a company is performing like the best in its class. Specialised consulting companies as well as industry associations can provide benchmarking services to access this vital information.
If benchmarking reveals that the company must improve its process performance, the company will either have to design its improved processes from scratch (re-engineering), or it would have to adopt the industry *best practice.* 'Best practice' is an umbrella word - it encompasses everything that can be done better whether it is the use of technology, improvement of processes, use of human skills and knowledge, or improved organisation. Therefore, to make 'best practice' more meaningful it shall be pointed out in subsequent chapters what best practice actually is from the point of view of the above components. In an aggregate manner, it is possible to identify the *presence* of best practice - based on performance measures, i.e. measurable characteristics of core pro-

cesses. However, to adopt 'best practice' one needs a deeper understanding of *why* and *how* best performing companies behave in order to achieved improved performance measures. This can only be done though a more detailed design process - or 'master planning' - that specifies all these components together, rather then in isolation. Merely adopting a new technology (e.g. because best performing companies use it) does not necessarily lead to successful imitation of best practice, because good performance is the result of the *combination* of the process, technology and organisation as well as individual human skills and knowledge.
Methods exist to assess the maturity level of the company from these points of view - using process assessment, the assessment of organisational performance, and the assessment of technology.

9.7.2 Core Competencies Selection and Development

An increasing interest in core competencies and capabilities has altered the balance of the 'make or buy' decision. Companies have been encouraged to buy items that they might previously have manufactured by themselves, with the intention of focusing on their core products and critical core competencies (Prahalad and Hamel , 1990). This question has become more important recently because of the increased ability of companies to co-operate with others. This ability to co-operate has both technological and organisational / social reasons:

- Information and communication technology has matured in the past two decades to the extent that reliable and fast information exchange does not need co-location of functions. Thus, production and service functions are more readily distributed among potentially remote participants;
- Standards (international and de-facto) have improved the compatibility of processes, allowing accurate product data exchange, electronic order processing and financial transactions based on generally accepted protocols;
- Companies have been forced to compete on a global market, optimising their development, production, delivery and service costs. An effective means to do so is to involve partners with optimised competencies in parts of the value chain;
- The quality movement (ISO 9000 series of standards as well as process improvement standards) have improved the predictability of business processes.

Core competencies comprise a set of skills and expertise that enable a company to deliver exceptional value to customers. They often involve an integrated system of capabilities that cannot easily be imitated by competitors (Hope and Hope , 1997).
According to Javidan (1998), the core competence hierarchy can be described as follows:

- Resources are the inputs into the organization's value chain. They can be categorised in three groups:
  - physical (technological) resources such as plant, equipment, location and assets;
  - human resources such as manpower, management team, training and experience;
  - organizational resources ,such as culture and reputation.
- Capabilities refer to the corporation's ability to exploit its resources. They consist of a set of business processes and routines that manage the interaction among its resources. Capabilities are functionally based (they are resident in a particular function), such as marketing capabilities, production capabilities, distribution and logistics capabilities. Example: Distribution Technologies of FedEx (e.g. Bar Coding), Information Technologies of Daimler-Benz;
- A competency is a cross-functional integration and coordination of capabilities. In a multi-business corporation, competencies are a set of skills and know-how housed in Strategic Business Units (SBUs). Example: Package tracking of FedEx, Cross-functional engineering knowledge in Daimler-Benz;
- Core Competencies cross SBU boundaries. They result from the interaction between different SBU competencies. Core competencies are skills and areas of knowledge that are shared across business units and result from the integration and harmonization of SBU competencies. A core competency is a collection of competencies that are widespread in the corporation. Example: Logistics in the case of FedEx, Engineering Excellence for Daimler-Benz.

The following activities could be used to identify and evaluate Core Competencies (Gallon et al. , 1995):

- Identify the enterprise core processes;
- Define the list of the enterprise capabilities in each core process. The classification of capabilities is represented in Table 9.2;
- Evaluate the capabilities and identify their strengths and weakness. This evaluation should be done according to strategic importance and relative / absolute strength;
- Identify core competencies. Valuation and identification of groups of capabilities that can be aggregated in order to form core competencies;
- Competencies evaluation: Evaluation of competencies in order to identify actual, desired and potential core competencies;
- Core Competencies Development Plan. Define development plans in order to sustain or improve actual core competencies and to develop or acquire new core competencies.

A Core Competencies development plan – and each of its components – is directed towards improving the competitive corporate position. Each plan

Table 9.2. Classification of Capabilities

Human Capital	Organization	Technological Capital
1) Knowledge	1) Communication level	1) Information systems
2) Abilities	2) Control level	2) Automated systems
3) Experience	3) Practices usage	3) Own technological capacities
4) Emotional Intelligence	4) Procedures / Instructions usage	4) Acquired technological capacities
5) Rational Intelligence	5) Standards usage	5) Manufacturing technologies
6) Personality	6) Techniques/methods usage	

component includes goals to be achieved, specific action programs, and intermediate targets against which interim performance can be measured. If a significant number of plan goals are achieved, corporate performance will improve and competitive position should improve. These goals should all lead toward improving the corporation's market value to book value ratio, which requires both solid growth and sound returns – a difficult combination to achieve (Gardner et al. , 1986).

According to Gardner et al. (1986), a set of performance measures may be used to measure performance and progress, including key factors for success and other important variables. Such performance measure should:

- Be measured at reasonable expense,
- Be controllable either directly (e.g. unit operating cost) or indirectly (e.g. market share),
- Be accounted for by a single identified executive or manager,
- Be readily understood,
- Cover the key elements of challenges in a given business,
- Indicate operating results as well as financial results,
- Include key comparisons with competitors,
- Be combined in groupings of not less than four or five, not more than eight to ten, so they can be comprehended as a group and provide effective overview of the business,
- Be reappraised or adjusted periodically to ensure that they continue to represent the key elements of challenges in a given business,
- Be supported by additional performance measures required down through the lower working level, thus aggregating performance measures between levels of management.

The identification of the required Core Competencies is a key for developing an enterprise concept based on the required human and technological capabilities to be a successful business.

9.7.3 Business Plan Preparation

A business plan is a written document that describes the future path of a business. A good business plan explains the business concept, summarizes the objectives of the business, depicts why the business will succeed., identifies the resources (both in terms of money, people and technologies) that will be needed by the business, describes how those resources will be obtained, and defines the process(es) involved to succeed.
Some benefits of producing a business plan include:

- Analysis of the business, recognition of opportunities and risks, and testing some of the assumptions;
- Identification of the capital needs for the business;
- Funding mechanism to find financial resources from banks and from investors;
- Communication means to tell employees, investors and others about the company's plans and strategies;
- A benchmarking tool to compare the progress and performance of the business.

It is a good idea for all businesses to prepare and regularly update their business plans. However, small businesses are most likely to prepare a business plan when they are just starting up or when a major change in their business is occurring (and often when additional investment, or a loan are needed). Some guidelines for preparing a good business plan are:

1. Determine clearly the objective of the business plan. Who is going to read the plan and what do we want them to do? The objectives can help the company decide how much emphasis to put on various sections of the business plan.
2. Allocate enough time and resources to thoroughly research the business plan. It is necessary to find out more about the industry, potential customers, potential competitors, and potential sales and costs.
3. Let someone else review the business plan It can be very useful to get feedback on the draft business plan from various people, including people associated with the business.
4. Write a unique business plan. The business plan should reflect what is important to the particular case and what key issues have to be addressed. One common mistake made by entrepreneurs is to borrow heavily from a sample business plan and simply change the names and some of the numbers.
5. Identify and address key points. Review the outline to ensure that all sections are consistent and that all the key issues have been addressed.
6. Make sure that the financial projections are realistic. The financial section is the most important section of the business plan because it identifies the company's financing needs and shows the profit potential of the business.

7. Write an outstanding Executive Summary. The keys to a good executive summary are that it should be short (2 pages at most), it should highlight what is important in the plan, and it should get the reader excited about the business described.

There are different on-line tools and business plan software that can help in producing a business plan, for example IBP Interactive Planner Starting (http://www.cbsc.org/ibp/), Costs Tool (http://www.bplans.com), or Business Plan Pro (http://www.paloalto.com/store/) It needs to be stressed that such tools will help defining a structure, but the content must be based on well researched points, as described above.
It is desirable to describe as part of the Business Plan as a step-by-step process, or Action Plan, where each element of the process is identified, together with its necessary inputs and the outcomes it must produce. To identify the activities in each step of this plan it is possible to use as a checklist the relations between Enterprise Entities, as described in the Business Model (such as normative and generative relations).
An example for a *normative* relation is where the specification of a desired set of products or services is known, and therefore determines the set of production tasks in a plant. An example of a *generative* relation is, where a plant develops the preliminary and detailed designs of a product or service. While normative relations usually require an analysis to be done, generative relations usually require the performance of design and implementation activities. The business plan must define the process that consists of these normative/analytic and generative/design-implementation activities.
Note that depending on the level of innovation required in the implementation of the Strategy, the Business Plan must take into account the need for one- or several iterations. Thus, while the types of activities in the Business Plan follow from the life-cycle relations among enterprise entities, the actual list of activities might include a decomposition of these into sub-activities – some of them repetitive / iterative (where the activity has high innovation content), and some of them one-off (where the activity is a routine one).
The level of innovation in the business plan cannot be measured in absolute terms: a task needs to be handled in the Business Plan as innovative if the people and technology available do not have the demonstrated capability to perform the given task.

9.8 Conclusion

The development of the enterprise concept is the elaboration of the mission and vision of the enterprise, and a means to define the strategy and plans to implement these. Since the change or design and implementation process (whether aiming at the creation of a new business or at the modification of an existing one) is likely to take considerable time, the concept needs to be defined

with a long term strategic view in mind. This Chapter has described activities and outcomes that are involved in turning a business idea (the identification of the business) into an operational plan that can be implemented.
Note that part of the concept is based on a long term view of the business, with a time horizon spanning a few years (mission, vision, strategies, core competency identification). Part of this view however consists of operationalising these concepts into objectives, plans, etc with a shorter time horizon (e.g. one year). For this reason, from time to time (e.g. yearly) the company needs to reassess these outcomes. Such reassessment needs to occur also in case there are significant technological, financial, market, or social events that might invalidate the assumptions based on which either part of the business concept was conceived.
If the time horizon of the given component was three years, then a re-evaluation of this component would have to take place every one or two years. The result of this re-evaluation may have flow-on effects, e.g. if strategy changes then the business plan would have to be re-evaluated and would be likely to undergo change.

References

Ackerman L.D. (2000). Identity is Destiny, San Francisco: Berret-Koehler.
Baghai M., Coley S., White D. (1999) The Alchemy of Growth, Reading: Perseus Books.
Belasco J.A., and Stead J.(1999). Soaring with the Phoenix, New York: Warner Books.
Bethlehem, PA. (1997). NGM Project, Next Generation Manufacturing: A Framework for Action, Agility forum.
Bhide A.(1999) The questions every Entrepreneur must answer, in Harvard Business Review on Entrepreneurship, Boston: Harvard Business School Press, pp. 1- 28.
Browne, J., Hunt, I., Zhang, J. (1999). The Extended Enterprise. Handbook of Life Cycle Engineering – Concepts, Models and Technologies (Edited by A. Molina, J.M. Sanchez, A. Kusiak). UK: Kluwer Academic Publishers.
Burgelman, M., Maidique A., Wheelwright S.C. (1995) Part II – Design and Evolution of Technology Strategy, in Strategic Management of Technology and Innovation, 2^{nd} edition, Chicago: Irwin, pp. 33 – 44.
Caskey, K.R. Co-operative Distributed Simulation and Optimization in Extended Enterprise (1995) IFIP WG 5.7 Working Conference on Managing Concurrent Manufacturing to Improve Industrial Performance, September 11-15, 1995, Seattle, Washington, USA.
Collin J.C., and Porras J.I. (1994). Built to last – Successful habits of visionary companies, New York: HarperBusiness.
David F.R. (1995). Strategic Management, New Jersey: Prentice Hall.

Drucker P.F (1985). Innovation and Entrepreneurship - Practice and principles, New York: Perenial Library, pp 150.

ENAPS (1999) ENAPS Model, http://www.enaps.com/public/public.htm, 1999

Gallon, Mark R., Stillman, Harold M, and Coates, D. (1995). Putting Core Competency Thinking into Practice, Research Technology Management. Industrial Research Institute, May – June 1995. pp. 20 – 28.

Gardner, J., Rachlin, R., Sweeny, A. (1986). Handbook of strategic planning, New York, John Wiley and Sons.

Gott, B. (1996). Empowered Engineering for the Extended Enterprise – A Management Guide, Cambridge, England, Cambashi Ltd.

Goggin, K., Jordan, P. Browne, J. (1997). Resource Sustainment – Issues and Opportunities, ME-SELA' 97, Loughborough University.

Grant, R.M. (2002). Contemporary Strategy Analysis - Concepts, Techniques and Applications, 4^{th} Edition, Oxford: Blackwell Publishers Ltd.

Hamel, Gary, and Heene, Aimé (1994). Competence-Based Competition. The Strategic Management Series. Hoboken, NJ: John Wiley & Sons.

Hamel, Gary and Prahalad, C.K. (1994). Competing for the Future. Harvard Business School Press.

Hill, T. (1989). Manufacturing Strategy, Homewood, IL: Richard D. Irwin.

Hoover G.(2001). Hoover's vision, New York: Texere.

Hope J., and Hope T. Competing in the Third Wave, Boston: Harvard Business School Press.

Javidan, Mansour (1998). Core Competence: What does it mean in practice? Long Range Planning, 31(1), pp 60-71. Great Britain: Elsevier Science.

Katzy, B.R., Schuh,G. (1999). The Virtual Enterprise, in Handbook of Life Cycle Engineering – Concepts, Models and Technologies (Edited by Molina,A., Sanchez,J.M. and Kusiak, A.). Great Britain: Kluwer Academic Publishers.

Kaplan R. and Norton D. (2001). The strategy focused organization, Boston: Harvard Business School Publishing Corporation, pp. 353.

Kotter J.P. (1998). What leaders really do, in Harvard Business Review on Leadership, Boston: Harvard Business School Press.

Levitt, T, 'Exploit the Product Life Cycle, Harvard Business Review, November-December 1965, pp. 81-94.

Miller, T.E., and Berger D.W. (2001). Totally Integrated Enterprises, Boca Raton: Raytheon Professional Services, LLC.

Mintzberg H., and Lampel J. (1999). Reflecting on the Strategy Process, Sloan Management Review, Spring 1999, pp. 21- 30.

Molina A., Ponguta S., Bremer C.F., Eversheim W. (1998). Framework for Global Virtual Business, Agility & Global Competition, Vol. 2, No. 3, pp. 56-69.

Molina Gutiérrez A., Flores Pineda M., Caballero Salinas D.M.(1999). Investigación de los Indicadores de Productividad y Mejores Prácticas de la Industria Manufacturera Regiomontana a través de un Estudio de Benchmarking,

International Journal of Mechanical Production Systems Engineering, No. 2, May 1999, pp. V-3, V-10 (in Spanish).

Moore, G.A. (1999). Crossing the Chasm – Marketing and Selling High Tech Products to Mainstream Customers, New York: Harper Business.

Mintzberg H. (1973). Strategy Making in Three Modes, California Management Review, 1973, pp. 44-53

Collins J. and Porras J. (1997) Built to last, New York: Harper Business, pp 20-237.

Osborne, D. and Plastrik, P. (1998). Banishing Bureaucracy - The five strategies for reinventing government, New York, Plume, Penguin Group.

Ostroff, F. (1999). The Horizontal Organization, New York: Oxford University Press.

PNUD, ONUDI (2000). Introducción a la Integración Productiva, in Spanish.

Porter, M. (1979). How Competitive Forces Shape Strategy, Harvard Business Review, 57, March-April 1979, pp. 86-93.

Porter, M. (1980). Competitive Strategy, New York: The Free Press.

Porter, M. (1997). From Competitive Advantage to Corporate Strategy, Harvard Business Review, May-June 1987, pp. 43 – 59.

Prahalad C.K. and Hamel G. (1990). The Core Competence of the Corporation, Harvard Business Review, May – June 1990, pp. 79 – 91.

Rea, P., and Kerzner H. (1997). Strategy Planning – A practical guide, New York: John Wiley & Sons.

Rehg J.A. (1994). Computer-Integrated Manufacturing, Prentice Hall Career & Technology.

Rodin R., and Hartman C. (1999). Free, Perfect and Now: Connecting to the Three Insatiable Customer Demands A CEO's True Story, New York: Simon & Schuster.

Shaker S.M., and Gembicki M.P. (1998). The War Room – Guide to Competitive Intelligence, New York: Prentice Hall.

Sternberg E. (1999). Chapter 1 Transformation: The Forces of Capitalist Change, in 21^{st} Century Economics – Perspectives of Socioeconomics for a World Changing World, Edited by W.E. Halal and K. Taylor, New York: St. Martin's Press, pp. 3-29.

Tang,V., Bauer, R. (1995). Competitive Dominance beyond strategic advantage and total quality management. New York: Van Nostrand Reinhold pp 137.

Zand, D.E. (1997) The Leadership Triad – Knowledge, Trust and Power, New York: Oxford University Press.

Appendix A – Business Plan Outline

1. Executive Summary Highlights
1.1 Objectives
1.2 Mission
1.3 Keys to Success

2. Company Summary
2.1 Company Ownership
2.2 Company History
2.2.1 Past Performance
2.2.2 Future Performance
2.3 Company Locations and Facilities

3. Products and Services
3.1 Product and Service Description
3.2 Competitive Comparison
3.3 Sales Literature
3.4 Sourcing
3.5 Technology
3.6 Service and Support
3.7 Future Products

4. Market Analysis Summary
4.1 Market Segmentation
4.2 Target Market Segment Strategy
4.2.1 Market Needs
4.2.2 Market Trends
4.2.3 Market Growth
4.3 Industry Analysis
4.3.1 Industry Participants
4.3.2 Distribution Patterns
4.3.3 Competition and Buying Patterns
4.3.4 Main Competitors

5. Strategy and Implementation Summary
5.1 Strategy Pyramids
5.2 Value Proposition
5.3 Competitive Edge
5.4 Marketing Strategy
5.4.1 Positioning Statements
5.4.2 Pricing Strategy
5.4.3 Promotion Strategy
5.4.4 Distribution Strategy

5.5 Sales Strategy
5.5.1 Sales by Year
5.5.2 Sales Forecast
5.5.2.1 Sales Monthly
5.5.2.2 Sales Forecast
5.5.3 Sales Programs
5.6 Milestones

6. Management Summary
6.1 Organizational Structure
6.2 Management Team
6.3 Management Team Gaps
6.4 Personnel Plan
6.5 Other Management Considerations

7. Financial Plan
7.1 Important Assumptions/ General Assumptions
7.2 Key Financial Indicators / Benchmarks
7.3 Break-even Analysis / Break-even Analysis
7.4 Projected Profit and Loss
7.5 Projected Cash Flow
7.6 Projected Balance Sheet
7.7 Business Ratios

10

ENTERPRISE MODELLING – THE READINESS OF THE ORGANIZATION

Ron Hysom

NCR *- (USA)

10.1 Introduction – The Enterprise Problem Space

Enterprises can be very confusing. They can be better understood if we look at them as groups of objects (i.e. people, tools, materials, etc.), which interact with each another. If we find that this approach is somewhat realistic, modelling an enterprise using this approach becomes more feasible. There are several important aspects of enterprises which, if understood, help us understand the enterprises themselves. The following discussion will address these important aspects. Figure 10.1 summarizes this discussion[1].
Let us use the symbol of a cube to examine these aspects. The primary cube is the Enterprise Cube. A set of secondary cubes expand upon aspects of the Enterprise Cube:

- The first aspect is perspective, represented by the Perspective Cube. One can describe the enterprise from different perspectives. Perspective is the orientation of the Enterprise Cube.
- The second aspect is life-cycle stage, represented by the cubes below the Enterprise Cube. There are different stages of the enterprise improvement life-cycle and each stage captures important, yet different knowledge about the enterprise.
- The third aspect is structure, represented by the Structure Cube. Structure is one of the primary dimensions of an Enterprise Cube.
- The fourth aspect is behaviour, represented by the Behaviour Cube. Behaviour is the second primary dimension of the Enterprise Cube.
- The fifth aspect is value, represented by the Value Cube. Value is the third primary dimension of the Enterprise Cube.

* National Cash Registers Company, http://www.ncr.com

[1] [editor's note] In the GERA modelling framework defined in Chapter 2 (and in all other modelling frameworks discussed in Chapter 3) a number of *modelling views* of the enterprise are introduced. As the this Chapter demonstrates, such enterprise models can be produced from different aspects.

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

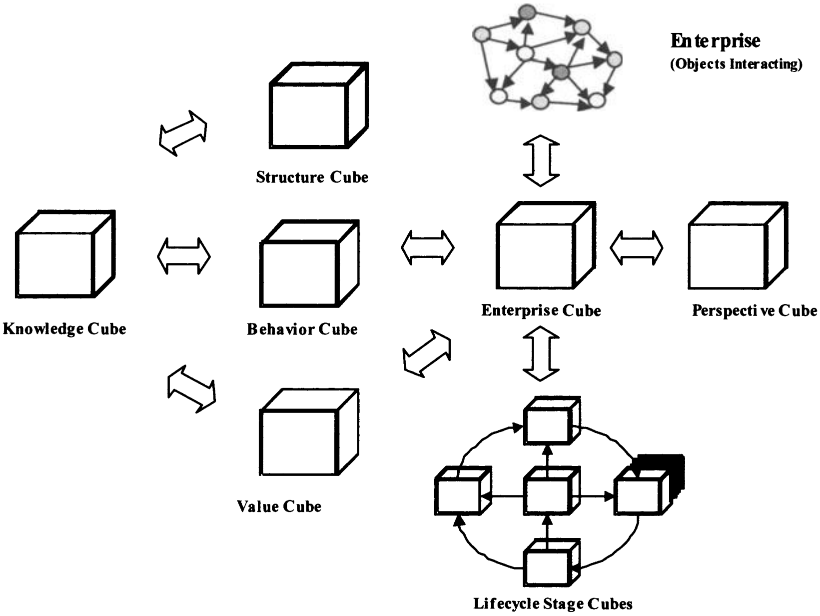


Fig. 10.1. Aspects of Enterprise Modelling

- The sixth aspect is knowledge, represented by the Knowledge Cube. All of the knowledge in all of the previous five cubes has attributes that are described by the Knowledge Cube.

No wonder enterprises seem complex. Let us get a little more specific.

10.2 Perspective Dimensions

One of the difficulties in understanding enterprises is that, whether we are consciously aware of it or not, we are usually working with at least three different perspectives: reality, perceptions and expectations (refer Fig. 10.2). The term 'reality' is used in this context to describe what is actually happening in an enterprise or what will actually happen under a given set of circumstances. Sometimes, this is called the *objective perspective* (i.e. without injecting personal opinion or bias).

Perceptions are subjective and personal concepts of what is believed to be happening in an enterprise. As can be readily appreciated, there are often distinct differences between reality and perceptions.

To make matters worse, expectations often enter into the description of what is happening in an enterprise. Expectations are also subjective and personal concepts, and bring in the desires of the individual rather than the beliefs of what is / or may happen.

If these different perspectives are not identified and separated when seeking to understand an enterprise, a very confused picture can emerge. Sometimes problems encountered in enterprises are caused primarily by people confusing these perspectives, and clarification can greatly improve the situation.

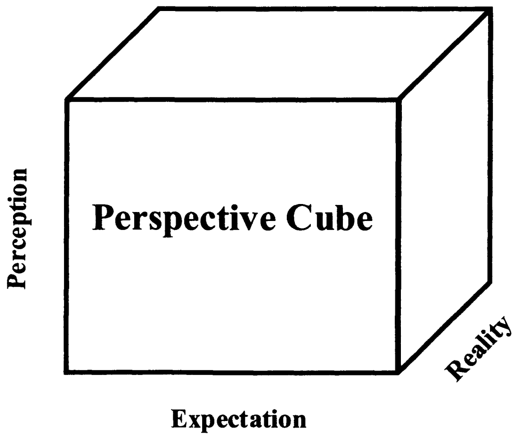


Fig. 10.2. Dimensions of Perspective

10.3 Stages of the Enterprise Improvement Life Cycle

There are several stages of the Enterprise Improvement Life-cycle, each capable of its own Enterprise Cubes (refer Fig. 10.3):

- As-Is – Representing the current enterprise. This stage derives its data primarily from the real world;
- Should-Be – Representing the enterprise as it should be (or the objective enterprise). This stage derives its data both from the expectations of enterprise stakeholders and the requirements of the environment;
- Needs-Are – Representing the gaps between the objective enterprise and the current enterprise. This stage derives its data from comparative analysis of the SHOULD-BE and AS-IS Enterprise models;
- Could-Be – Representing possible alternative enterprises. This, and the TO-BE stages derive their data from the experience of the enterprise designers and from models;
- To-Be – Representing the preferred enterprise from all the COULD-BE alternatives. This becomes the target enterprise.

For this reason, each of the cubes discussed in Sections 10.4-10.7 could potentially be constructed for each life-cycle stage.

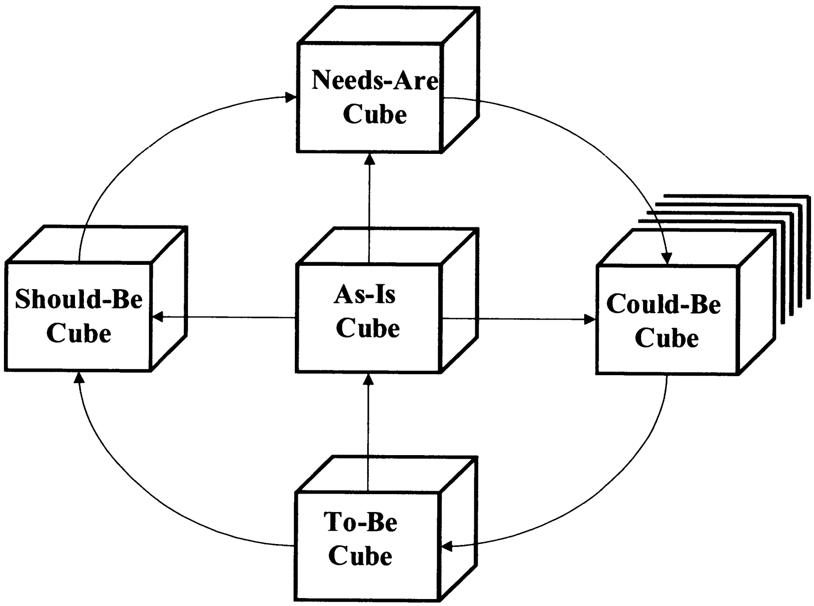


Fig. 10.3. Stages in the enterprise improvement life-cycle

10.4 Enterprise Dimensions

There are at least three major aspects of an Enterprise, all of which are interrelated and understanding these should improve the Enterprise (refer Fig. 10.4). The degree to which each aspect is considered in a Enterprise can range quite widely:

- Structure – What are the objects and how do they relate to each other? This is the most common and easiest aspect examined. Most techniques used to study enterprises examine structure. The degree to which structure is considered can range from very simple (very common but not very realistic) to quite complex (unusual but more realistic).
- Behaviour – How does this structure respond under different circumstances? This aspect is typically the one mentioned in SHOULD-BEs. It is more difficult to examine, however, particularly in COULD-BEs and TO-BEs. The degree to which behaviour is considered can range from static (not very realistic, but the usual degree examined) to dynamic (much more difficult, but also much more realistic).
- Value – Is this behaviour really adding value to the enterprise? This aspect is also mentioned sometimes in SHOULD-BEs, but is very difficult to examine in the other types. The degree to which value is considered can range from unknown (value is not understood when examining it)

to understood (a rich approach is used to determine value-added when examining it).

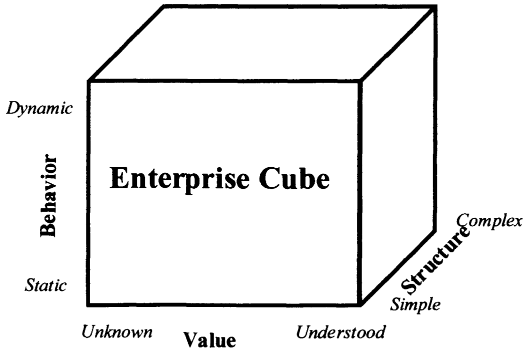


Fig. 10.4. Dimensions of the Enterprise

The effectiveness of enterprise improvement is greatly dependent upon understanding the structure, behaviour and value of any possible changes (and to add to the challenge, they are all related). Structure is the easiest to control. Behaviour is greatly dependent upon structure. Improvements can occur by simply changing the structure. However, behaviour is difficult to predict by simply examining the structure of an enterprise. The linkage between structure and behaviour is often not fully understood, so behaviour may have to be observed by activating the structure over time[2].
Both behaviour and structure are interesting and helpful, but if no consideration is given to the value added by making changes in both, change can become an exercise of futility. The enterprise can actually be damaged by change that does not add value.
It is important to understand all three aspects to make effective improvements to an enterprise.

10.5 Structure Dimensions

The structure aspect of an enterprise, ranging from simple to complex, can also be viewed as consisting of at least three aspects (refer Fig. 10.5). Each of the aspects of structure involve a range of degrees:

- Objects – which objects (and their characteristics) are involved in the enterprise? The degree to which objects can be modelled can range from none (simple) to an unlimited number of objects (complex);

[2] [editor's note] e.g. using simulation.

- Location – where are these objects located? The degree of importance of location can range from a local situation (simple) to the need to understand location in a global context (complex).
- Connectivity – how are the objects logically or physically connected (i.e. related) in the enterprise? The degree of connectivity is important from isolated objects, which do not influence each other (simple), to complete connectivity where all objects have important relationships to one another (complex).

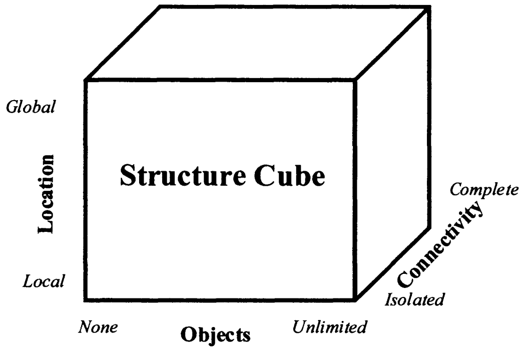


Fig. 10.5. Dimensions of Structure

To understand the structure of the enterprise, it is important to understand the objects involved in the enterprise, the locations of those objects and the connectivity between those objects. Objects, location and connectivity are all related.

Objects are resources, which can be deployed in an infinite variety of ways. They possess certain capabilities which make them useful in accomplishing goals. Because of this, it is necessary to understand all important objects and their characteristics.

Often objects are only able to play an important role in an enterprise if they are co-located; it is therefore important to understand where the objects are located and if necessary to move them for effectiveness.

Both objects and their locations are important, but without an understanding of the connectivity or relationships between objects, the structure (and therefore the roles the objects play) will not be understood. Ultimately however, the roles an object plays in an enterprise will determine how it performs.

It is therefore important to understand all three aspects of structure in order to make effective improvements to an enterprise.

10.6 Behaviour Dimensions

The behaviour aspect of an enterprise[3], ranging from static to dynamic, can also be viewed as consisting of several (sub)aspects (refer Fig. 10.6). The importance of each of the aspects of behaviour can also exhibit a range of degrees:

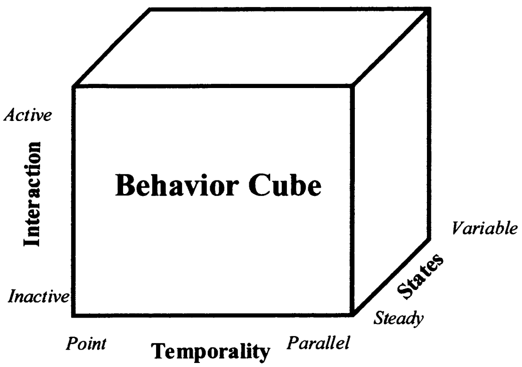


Fig. 10.6. Dimensions of behaviour

- States – The behaviour of an object can be better understood if defined when in different states. What does the object do while it is within each valid object state and what triggers transitions between object states? This is not always obvious and not particularly simple to understand, yet ultimately governs how the object can behave. The importance of object states can range from ignoring states or assuming a steady state (static) to understanding variable states and their transitions (dynamic).
- Interaction – Under what circumstances do the individual objects interact? When they do interact, how do they affect the state of each object? This is also difficult to envision, but is critical for understanding why the overall enterprise behaves the way it does. The degree to which interaction is important can range from inactive (with the detail interaction immaterial (static)) to active interaction (dynamic).
- Temporality – How do the objects behave over time? Intuitively, this is the easiest to understand at a higher level, given a real-life enterprise or a model to examine. But when many objects interact in parallel, this aspect can become very difficult to understand. The degree to which temporality is important ranges from examining the enterprise at a point in time (static) to evaluating its behaviour as a result of many objects interacting in complicated temporal relationships in parallel (dynamic).

[3] which was one axis of the enterprise cube (refer Fig. 10.4)

Objects exhibit certain behaviour while they are in specific states. Which state an object is in is often dependent upon interaction with other objects. That interaction will frequently change the object state and will therefore impact the behaviour of the object. The overall behaviour of the interaction (i.e. the enterprise) will therefore be the result of the cumulative behaviour of the objects.
Both state life cycles and interaction scenarios are important, but if no consideration is given to the type of temporal relationships and their change over time, then perspective is gained for only a single point in time and that perspective is not very helpful in dealing with reality.
To understand the behaviour of the enterprise, it is important to understand both the state life cycles of the objects involved in the enterprise and the interaction of those objects over time. Temporality, states and interaction are all related.
It is therefore important to understand all three aspects of behaviour to make effective improvements to an enterprise.

10.7 Value Dimensions

The value aspect of an enterprise, ranging from *unknown* to *understood*, can also be viewed as consisting of at least three aspects; risk, cost, and capability. Thus, in a similar way to the behaviour aspect of the enterprise cube (Fig. 10.4) the value aspect can also be further detailed in the value cube (refer Fig. 10.7). Value can be defined as the capability for the cost ('bang for the buck') given a certain level of risk. Each of the value aspects can also be important in a range of degrees:

- Capability - how is the capability of the objects improved in the enterprise? The degree to which capability is important can range from ignored (unknown) to considered and understood, i.e. competent;
- Cost - how much cost is incurred by the objects while participating in the enterprise? The degree to which cost is important ranges from disregarded (unknown) to controlled (understood), where the cost of each object is controlled throughout the enterprise;
- Risk - how is the cost and capability exposed to adverse impacts by threats and to what extent is the exposure mitigated by effective controls? The degree to which risk is important ranges from naive (unknown) to mitigative (understood) where the risks are identified and realistic controls are used to reduce the exposure.

To understand the value aspect of the enterprise, it is important to understand both the capabilities and the cost of the objects involved in the enterprise as well as the effect of adverse risks against those objects. Capability, cost and risk are all related.

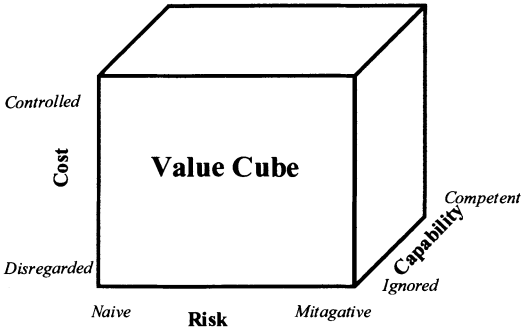


Fig. 10.7. Dimensions of value

Objects possess certain capabilities. Whether we realise it or not, underlying the fabric of enterprises is the intention of *changing capabilities.* When capabilities are such that others desire them, a market is created and business thrives. When capabilities are no longer desired, business ceases. It is therefore important, perhaps even critical, that capability is understood.
As objects are used to change capabilities, cost is incurred. Value is added when the capability is increased for the same or less cost, or the same capability is maintained for less cost, or both occur at the same time. Cost must be understood.
Both capability and cost are important, but if no consideration is given to the risk of loss of capability or increase of cost from adverse factors, then the realistic value added will remain unknown.
It is therefore important to understand all three aspects of value to make effective improvements to an enterprise.

10.8 Knowledge Dimensions

How well we understand any aspect of an enterprise is dependent upon the knowledge we can extract about the enterprise. The validity of enterprise knowledge is affected by at least four aspects: completeness, predictability, precision and certainty. Three of these aspects of knowledge (completeness, predictability and precision) determine the amount of knowledge acquired about the enterprise. The fourth aspect (certainty) determines the *reliability* of the knowledge. Note that these knowledge dimensions (refer Fig. 10.8) apply to all previous aspects (such as behaviour, structure and value) – specifically to their dimensions – thereby creating a multi-dimensional space of enterprise modelling.

- Completeness – how complete is the knowledge about an enterprise? What we do not know may be more important than what we do know! Complete-

ness impacts the sufficiency of knowledge. The completeness of knowledge can range from insufficient to necessary & sufficient.

- Predictability – what happens in enterprises is probabilistic in nature. What do we know about the stochastic behaviour of the objects involved? Predictability impacts the consistency of knowledge. The predictability of knowledge can range from 0% probable (impossible)[4] to 100% probable (deterministic). Everything in-between falls into the stochastic category.
- Precision —how precise is the knowledge about an enterprise? Terms often used to describe knowledge are very 'fuzzy' in meaning. Do we really understand what is meant by the terms? Precision impacts the meaning of knowledge. The precision of knowledge can range from vague (meaning is different for each party involved) to meaningful (meaning is precisely understood by all parties involved).
- Certainty – finally, for the knowledge we have and understand, to what extent is that knowledge actually true? What is the evidence supporting the knowledge? How much can we believe to be useful? Even though we may have sufficient (completeness) and probable[5] (predictability) knowledge whose meaning is commonly understood (precision), we may not be certain that the knowledge is reliable enough to be used to understand the enterprise.

Certainty is a factor of belief that is based upon evidence that a piece of knowledge is true or not true. Certainty can be thought of as a two-dimensional concept, with certainty that something is *true* as one dimension and the certainty that something is *false* (or not true) as the other. *Ignorance* lies at the origin (no certainty of either true or false). *Contradiction* lies at the intersection of true and false.
Uncertainty lies along the dotted line linking *ignorance* to *contradiction.* The probability that something is true lies along the solid line linking *true* with *false.*
Whether we recognize it or not, each item of knowledge carries with it a degree of certainty. It is this degree of certainty that determines, often unconsciously, whether we believe that the knowledge is true enough to be used to form concepts or to make decisions.
Any or all of the four aspects of knowledge (completeness, predictability, precision, and certainty) are often ignored when working with enterprise knowledge. If these aspects are not considered, or techniques are used that are incapable

[4] [editor's note] To say that '0% probable = impossible' is mathematically speaking not exactly true (nor does '100% probable = deterministic'), but for this treatment this is a legitimate simplification. What matters here is how certain the modeller is about the facts captured in the enterprise models.

[5] [editor's note] Probable means here that in light of the evidence the model gives predictable results. However, this does not necessarily mean that the modeller is necessarily *certain*about the facts based on which the model was developed. Therefore the 'certainty' factor is introduced.

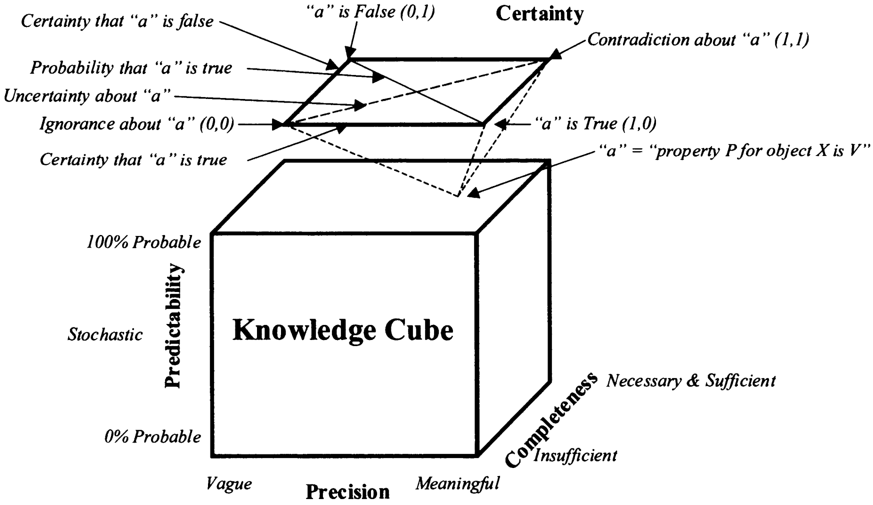


Fig. 10.8. Dimensions of Knowledge

of dealing with these aspects, then unrealistic and even invalid conclusions can be derived without realizing it and devastating results could occur. However, all four of these aspects could be considered by using techniques capable of addressing them and decision-makers could work with more realistic knowledge when making enterprise decisions.

10.9 Assessment of the Capability to Use Enterprise Modelling Technology

To effectively model something as complex as an enterprise, a modelling technology is needed. There are many types of technologies which have been- and could be used to model various aspects of an enterprise. The real challenge is to find technology that can handle those aspects that you deem important and to do that in a consistent manner. To help assess the capability of enterprise modelling technology, let us look at what is involved in the enterprise modelling process (refer Fig. 10.9). Any useful technology needs to assist in most, if not all seven of these stages.

This generic process involves seven stages. The titles should be self-explanatory – at least to explain generally what happens in each stage. When reviewing the assessment criteria for each stage, what needs to be considered in that stage should become clearer. In addition to the technical criteria for each of the seven stages, there are criteria for manageability, usability and deployability. A point system is suggested in Fig. 10.10, but the evaluator is

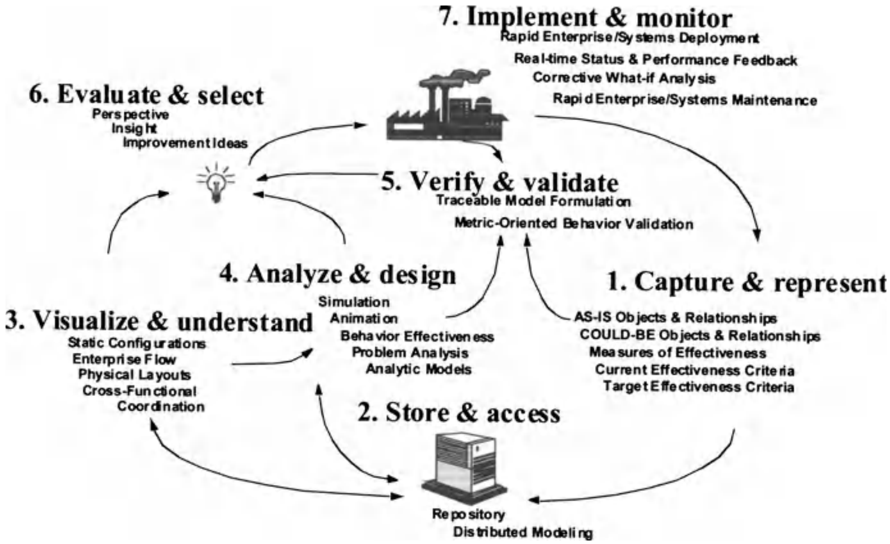


Fig. 10.9. The enterprise modelling process

free to assign points as needed, while Fig. 10.11 summarizes the total points possible for this capability assessment.
It is recommended that, when beginning to assess enterprise modelling technology, the modeller should initially determine the required levels of capability for the modelling effort. This establishes the baseline of required capabilities. Then, as each technology is assessed, it can be compared to the baseline. An overview graph is shown in Fig. 10.12 and Fig. 10.13 to plot an assessment.

Point Value	Capability Assessment Level
0	None - No capability
1	Weak - Barely useful capability
2	Good - Useful capability
3	Strong - Significantly useful capability
4	Excellent - World Class capability

Note: The meaning attached to each assessment level for each criteria differs and is therefore described for each criteria. The values for each assessment level are the same throughout the form.

Fig. 10.10. Point scale for each capability

The following capability categories are addressed in this assessment form:

Code	Capability Assessment Category	Number of Criteria	Total Possible Points
T1	Technical - 1. Capture & represent	17	68
T2	Technical - 2. Store & access	5	20
T3	Technical - 3. Visualize & understand	5	20
T4	Technical - 4. Analyze & design	8	32
T5	Technical - 5. Verify & validate	5	20
T6	Technical - 6. Evaluate & select	4	16
T7	Technical - 7. Implement & monitor	4	16
M	Manageability	5	20
U	Useability	4	16
D	Deployability	9	36
	Totals	66	264

Fig. 10.11. Point system for capability categories

10.9.1 T1 – Technical Assessment Criteria – Capture and Represent

T1.1 Capability to Model Enterprise Entities

None	Cannot represent enterprise entities in a model.
Weak	Can represent 'canned' entities, but unable to tailor entities to meet unique needs.
Good	Supports the tailoring of entities, including adding new entities and attributes and changing existing attributes.
Strong	In addition, provides support for defining entity types/classes and static entity instantiations which can be used to model structure.
Excellent	In addition, provides support for defining dynamic entity instantiations which can be used to model behaviour.

T1.2 Capability to Model Entity Relationships

None	Cannot represent any relationships between entities.
Weak	Can represent relationships, but little additional relationship support.
Good	In addition, supports the tailoring of relationships, including adding new relationships and attributes and changing existing attributes..
Strong	In addition, provides support for defining relationship types/classes and static relationship instantiations which can be used to model structure.
Excellent	In addition, provides support for defining dynamic relationship instantiations which can be used to model behaviour

T1.3 Capability to Capture Enterprise Data From All Necessary Sources

None	Cannot capture any enterprise data.
Weak	Supports manual entry of data using keyboard and/or mouse.
Good	In addition, supports importing of bit-mapped data from other sources.
Strong	In addition, supports importing of identified enterprise entities from other sources.
Excellent	In addition, supports linking of externally-located data to internal enterprise entities..

T1.4 Capability to Model All Stages of Enterprise Improvement Life-cycle

None	Cannot model any aspects separately.
Weak	Can build stand-alone models (current AS-IS, required SHOULD-BE, gaps NEEDS-ARE, alternative COULD-BEs, or preferred TO-BE).
Good	Can display any combination of above model stages in same model.
Strong	In addition, can build transition phases between any of the above stages in same model.
Excellent	In addition, can establish and trace logical links between any portion of any stage in same model

T1.5 Capability to Model Completeness of Enterprise Knowledge

None	Cannot represent or determine completeness of enterprise knowledge.
Weak	Can represent enterprise general opinion of completeness, but little additional completeness support.
Good	In addition, provides support for determining what enterprise knowledge is necessary in a specific model.
Strong	In addition, provides support for determining what enterprise knowledge is sufficient in a specific model.
Excellent	In addition, provides support for quantifying degree of completeness of enterprise knowledge.

T1.6 Capability to Model Predictability of Enterprise Knowledge

None	Cannot represent or determine predictability of enterprise knowledge.
Weak	Can represent general opinion of predictability, but little additional predictability support.
Good	In addition, provides support for determining what enterprise knowledge is stochastic in nature in a specific model.
Strong	In addition, provides support for deriving the probability factors for each affected item in a specific model.
Excellent	In addition, provides support for quantifying degree of predictability of enterprise knowledge.

T1.7 Capability to Model Precision of Enterprise Knowledge

None	Cannot represent or determine precision of enterprise knowledge.
Weak	Can record vague expressions of enterprise knowledge, but little additional precision support.
Good	In addition, provides support for defining ranges of possible meanings for vague expressions.
Strong	In addition, provides support for handling vague enterprise knowledge including ranges of meanings, algorithms for performing logical functions, etc.
Excellent	In addition, provides support for quantifying meanings of the use of vague enterprise knowledge.

T1.8 Capability to Model Certainty of Enterprise Knowledge

None	Cannot represent or calculate certainty of enterprise knowledge.
Weak	Can represent general opinion of certainty, but little additional certainty support.
Good	In addition, provides support for calculating the certainty factors for each entity affected by other certainty factors.
Strong	In addition, provides support for ignoring selected certainty factors when deriving calculated certainty factors.
Excellent	In addition, provides support for quantifying degree of certainty of each affected item of enterprise data.

T1.9 Capability to Model Entity Locations

None	Cannot distinguish the location of an entity.
Weak	Can distinguish an entity's initial location, but little additional locational support.
Good	In addition, provides support for changing location based upon specified criteria.
Strong	In addition, provides support for tracking the change of locations and the triggers for initiating the change.
Excellent	In addition, provides support for identifying distance between locations and time spent traversing between locations.

T1.10 Capability to Model Entity States

None	Cannot distinguish between separate entity states.
Weak	Can distinguish separate entity states, but little additional state support.
Good	In addition, provides support for defining activity within each entity state and links to interactions for initiating transitions between states.
Strong	In addition, provides support for identifying specified sequences of states for entities.
Excellent	In addition, provides support for quantifying volume and type of entities occupying each state and role.

T1.11 Capability to Model Entity Interactions

None	Cannot represent entity interactions.
Weak	Can represent general entity interactions, but little additional interaction support.
Good	In addition, provides support for describing the conditions within which interactions occur and the entity states which trigger interactions.
Strong	In addition, provides support for linking interactions to entity state sequences.
Excellent	In addition, provides support for quantifying degree, volume and type of interactions.

T1.12 Capability to Model Entity Temporality

None	Cannot represent any temporal aspects.
Weak	Can represent general sequencing of operations, but little additional temporal support.
Good	In addition, provides support for representing passing of time (duration) and the generation of entity instantiations according to a user-specified pattern at discrete time intervals and within specified time windows.
Strong	In addition, provides support for representing temporal relationships (i.e. precedes, synchronizes, meets, laps, spans, begins and finishes) between entity interactions.
Excellent	In addition, provides support for quantifying time spent by each entity in each state, state sequence and interaction.

T1.13 Capability to Model Entity Capability

None	Cannot represent entity capability.
Weak	Can represent entity as capable of specific tasks, but little additional capability support.
Good	In addition, provides support for specifying levels of capability with associated impact on interaction parameters.
Strong	In addition, provides approach for identifying impact of the level of entity capability on the level of capability of entities produced or changed by an interaction.
Excellent	In addition, provides support for quantifying degree of entity capability as entity progresses through its life-cycle.

T1.14 Capability to Model Entity Cost

None	Cannot represent entity cost.
Weak	Can represent unit cost of entity, but little additional cost support.
Good	In addition, provides support for deriving cost of specific interactions with an entity.
Strong	In addition, provides support for identifying impact of the cost of enabling entities on the cost of entities produced or changed by an interaction.
Excellent	In addition, provides support for quantifying cost as entity progresses through its life-cycle.

T1.15 Capability to Model Enterprise Risk

None	Cannot represent risk in a enterprise.
Weak	Can represent simple risk probability factors, but little additional risk management support.
Good	In addition, provides support for representing sources of exposure (threats) with associated risk factors and entities at risk.
Strong	In addition, provides support for identifying impact of the level of risk of sources of exposure on the level of capability and cost of entities produced or changed by an interaction as well as the specific controls, with associated capability and cost parameters, which would mitigate the level of exposure.
Excellent	In addition, provides support for quantifying degree of risk as entity progresses through its life-cycle

T1.16 Capability to Perform Abstraction of Model

None	Cannot abstract different refined models into a common higher-level model.
Weak	Can abstract simple hierarchical partitions using the same formalism into a common higher-level model, but little additional abstraction support.
Good	In addition, provides support for abstraction based upon user-specified criteria.
Strong	In addition, provides support for flexible abstraction of refined models using different formalisms based upon user-specified criteria and identifying degree of abstraction.
Excellent	In addition, provides support for automatic model abstraction of refined models using different formalisms based upon user-specified parameters with support for identifying degree of abstraction.

T1.17 Capability to Perform Refinement of Model

None	Cannot refine an abstracted model into a set of lower-level models.
Weak	Can refine an abstracted model into 'black-box' models using the same formalism, but little additional refinement support.
Good	In addition, provides support for 'clear-box' refinement of model elements based upon user-specified criteria.
Strong	In addition, provides support for flexible refinement of abstracted models into lower-level models using different formalisms based upon user-specified criteria and identifying degree of refinement.
Excellent	In addition, provides support for automatic model refinement of abstracted models into lower-level models using different formalisms based upon user-specified parameters with a rich set of metrics identifying degree of refinement.

10.9.2 T2 – Technical Assessment Criteria – Store and Access

T2.1 Capability to Maintain Data Consistency

None	Cannot maintain data in a consistent manner.
Weak	Can maintain data consistency within a model, but little additional data consistency support.
Good	In addition, provides support for controlling consistency of data types across models.
Strong	In addition, provides support for maintaining a common glossary for defining data across models.
Excellent	In addition, provides support for automatic checking of consistency across models as data is entered.

T2.2 Capability to Manage Model Partitioning

None	Cannot segment model into different partitions.
Weak	Can segment model into simple partitions based upon hierarchy (hierarchy-based extract), but little additional partitioning support.
Good	In addition, provides support for partitioning based upon user-specified criteria (criteria-based extract).
Strong	In addition, provides support for distributing and tracking partitions across network while maintaining logical integrity.
Excellent	In addition, provides support for automatic physical model partitioning, distribution and tracking across network based upon user-specified parameters, model size and load with support for identifying degree of partitioning and quantifying model load by platform.

T2.3 Capability to Manage Model Interfacing

None	Cannot specify interfaces between models.
Weak	Can specify interfaces between models by using textual or graphical annotations, but little additional interfacing support.
Good	In addition, provides support for logically linking entities in physically separate models, but models cannot act as one logical model.
Strong	In addition, provides support for logically linking entities in physically separate models in such a way as to make the separate models act as one logical model.
Excellent	In addition, provides support for automatic physical model traversal based upon user-specified parameters with support for collecting modelling data for reporting and/or analysis.

T2.4 Capability to Manage Conflict Resolution Between Models

None	Cannot identify any conflicts between different models.
Weak	Can identify conflicts between different models using the same formalism, but little additional conflict resolution support.
Good	In addition, provides support for resolution of those conflicts.
Strong	In addition, provides support for identifying and resolving conflicts between models with different formalisms and providing the degree of resolution.
Excellent	In addition, provides support for conflict resolution across distributed models of any formalism and support for quantifying resolution parameters by model and platform.

T2.5 Capability to Manage Model Integration

None	Cannot integrate different models into a common model.
Weak	Can integrate simple hierarchical partitions using the same formalism into a common model, but little additional integration support.
Good	In addition, provides support for integration based upon user-specified criteria.
Strong	In addition, provides support for flexible integration of distributed model partitions using different formalisms based upon user-specified criteria while maintaining logical integrity and identifying degree of integration and inclusion by platform.
Excellent	In addition, provides support for automatic physical model collection and integration of distributed model partitions using different formalisms based upon user-specified parameters, model size and load with support for identifying degree of integration and inclusion and quantifying model load by platform.

10.9.3 T3 – Technical Assessment Criteria – Visualize and Understand

T3.1 Capability to Tailor Representations Used in Model

None	Cannot tailor representation of entities and relationships within model.
Weak	Can change size, color and line type for entities and relationships, but little additional tailoring support.
Good	In addition, provides support for changing graphically the representation patterns used for entity types or classes.
Strong	In addition, provides support for defining/importing complex bit-mapped patterns used for entity types and individual entities.
Excellent	In addition, provides support for dynamically changing representations based-upon data and/or conditions within the model.

T3.2 Capability to Control Manipulation of Entities in Model

None	Cannot control manipulation of entities and relationships within model (i.e. read-only presentation).
Weak	Can manually perform standard copy, cut, paste and move operations on entities and relationships, but little additional manipulation support.
Good	In addition, provides support for changing size of entities and zooming-in and out to visualize different depths of the model.
Strong	In addition, provides support for selectively hiding and revealing selected entities and relationships to visualize various model aspects of the model.
Excellent	In addition, provides support for defining visual layout guidelines and automatically manipulating entities based-upon these guidelines.

T3.3 Capability to Control Printed Output of Model

None	Cannot produce printouts of model..
Weak	Can produce 'canned' printouts of model, but little additional printing support.
Good	In addition, provides support for selecting portions of the model for printing.
Strong	In addition, provides support for specifying size, scaling, layout, color and printer format of the printout.
Excellent	In addition, provides support for producing a wide-range of printout formats to route to any selected output device.

T3.4 Capability to Tailor Visual Controls of Model Behaviour

None	Cannot control activation of model.
Weak	Can control activation of model with 'canned' controls, but little additional control support.
Good	In addition, provides support for specification and tailoring of controls to meet user-specified requirements.
Strong	In addition, provides support for tailoring of 'user-friendly' controls with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for dynamically changing control of model activation in real-time based-upon data and/or conditions within the model.

T3.5 Capability to Tailor Visualization of Model Behaviour

None	Cannot visualize activation of model.
Weak	Can visualize activation of model with 'canned' visualization approaches, but little additional visualization support.
Good	In addition, provides support for specification and tailoring of visualization to meet user-specified requirements.
Strong	In addition, provides support for tailoring of 'user-friendly' visualization with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for dynamically changing visualization of model activation in real-time based-upon data and/or conditions within the model.

10.9.4 T4 – Technical Assessment Criteria – Analyse and Design

T4.1 Capability to Construct Analysis Model

None	Cannot convert a constructed model into an analysis model.
Weak	Can provide simple download of constructed model to an analysis tool, but little additional construction or conversion support.
Good	In addition, provides support for converting constructed model to an analysis model requiring additional modeller intervention before an analysis can be run..
Strong	In addition, provides rich set of capabilities designed to minimize the effort and time necessary to convert the analysis model.
Excellent	Provides automatic conversion or building of analysis model from the constructed model without modeller intervention.

T4.2 Capability to Analyse Model Structure (Static Analysis)

None	Cannot support analysis of a model for diagnosis of structure.
Weak	Can support structural analysis of a model with one 'canned' analysis aspect (i.e. integrity, consistency, connectivity, critical path, etc.),, but little additional analysis support.
Good	In addition, provides support for specification and tailoring of analysis approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one analysis approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing analysis approaches in real-time based-upon data and/or conditions within the model. (structural self-analysis)

T4.3 Capability to Analyse Model Behaviour (Dynamic Analysis)

None	Cannot support analysis of a model for diagnosis of behaviour.
Weak	Can support behavioural analysis of a model with one "canned' analysis approach (i.e. calculation, simulation, inferencing, animation, etc.),, but little additional analysis support.
Good	In addition, provides support for specification and tailoring of analysis approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one analysis approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing analysis approaches in real-time based-upon data and/or conditions within the model. (behavioural self-analysis)

T4.4 Capability to Perform Discovery in Models

None	Cannot perform discovery in a model to gain insight into principles, relationships, etc that may be causing certain behaviour.
Weak	Can perform discovery in a model with one 'canned' discovery approach, but little additional discovery support.
Good	In addition, provides support for specification and tailoring of discovery approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one discovery approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing discovery approaches in real-time based-upon data and/or conditions within the model. (self-diagnosis)

T4.5 Capability to Configure Models

None	Cannot support configuration (selection of a subset of entities & relationships) of a model to meet certain requirements.
Weak	Can support configuration of a model with one 'canned' configuration approach, but little additional configuration support.
Good	In addition, provides support for specification and tailoring of configuration approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one configuration approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing configuration approaches in real-time based-upon data and/or conditions within the model. (self-configuring)

T4.6 Capability to Design Models

None	Cannot support design of a model to meet certain behavioural requirements.
Weak	Can support design of a model with one 'canned' design approach, but little additional design support.
Good	In addition, provides support for specification and tailoring of design approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one design approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing design approaches in real-time based-upon data and/or conditions within the model. (self-designing)

T4.7 Capability to Adjust Models

None	Cannot support adjustment of a model to improve behaviour.
Weak	Can support adjustment of a model with one 'canned' adjustment approach, but little additional adjustment support.
Good	In addition, provides support for specification and tailoring of adjustment approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one adjustment approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing adjustment approaches in real-time based-upon data and/or conditions within the model (self-healing).

T4.8 Capability to Optimise Models

None	Cannot support optimization of a model to gain optimum or sub-optimum behaviour.
Weak	Can support optimisation of a model with one 'canned' optimization approach, but little additional optimization support.
Good	In addition, provides support for specification and tailoring of optimization approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one optimization approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing optimization approaches in real-time based-upon data and/or conditions within the model (self-optimizing).

10.9.5 T5 – Technical Assessment Criteria – Validate and Verify

T5.1 Capability to Assess the Appropriateness of Formalism

None	Cannot assess the appropriateness of any formalism for meeting specific modelling goals.
Weak	Provides support for understanding a set of specific modelling goals, but little additional appropriateness assessment support.
Good	In addition, provides support for identifying aspects of a target enterprise which must be modeled by a prospective formalism to meet modelling goals and the capabilities of the prospective formalism.
Strong	In addition, provides support for identifying aspects of a enterprise which cannot be addressed by a prospective formalism and a method for determining degree of appropriateness of the formalism for meeting modelling goals.
Excellent	In addition, provides support for quantifying the appropriateness of any formalism for meeting any set of modelling goals.

T5.2 Capability to Assess the Range of Applicability of Formalism

None	Cannot assess the range of applicability of any formalism in adequately modelling a specific enterprise.
Weak	Provides support for understanding the range of modelling constraints present in a particular modelling situation, but little additional appropriateness assessment support.
Good	In addition, provides support for identifying the range of capabilities of a prospective formalism on a specified tool and platform.
Strong	In addition, provides support for identifying aspects of a enterprise modelling situation which cannot be addressed adequately by a prospective formalism on a specified tool and platform and a method for determining degree and range of applicability of the formalism for modelling a specific enterprise.
Excellent	In addition, provides support for quantifying the range of applicability of any formalism on a specified tool and platform for meeting any modelling situation and, if inadequate, support for determining what formalism, tool and platform will be provide an adequate capability for the situation.

T5.3 Capability to Verify Model Construction

None	Cannot verify a constructed model to its sources of data.
Weak	Supports manual review of a constructed model with its data sources, but little additional verification support.
Good	In addition, provides support for capturing review comments (redlining) and comparing them to constructed model.
Strong	In addition, provides support for selected parameter verification checks & identifying differences between the constructed model and its data sources.
Excellent	In addition, provides support for quantifying degree of verification of a constructed model to its data sources.

T5.4 Capability to Validate Model Structure

None	Cannot validate the structure of a constructed model to the actual enterprise.
Weak	Supports manual review of the structure of a constructed model with enterprise owners, but little additional validation support.
Good	In addition, provides support for capturing review comments (redlining) and comparing them to constructed model.
Strong	In addition, provides support for selected parameter validation checks & identifying differences between the structures of a constructed model and the actual enterprise.
Excellent	In addition, provides support for quantifying degree of structural validation of a constructed model to the actual enterprise.

T5.5 Capability to Validate Model Behaviour

None	Cannot validate the behaviour of a constructed model to the actual enterprise.
Weak	Supports manual review of the behaviour of a constructed model with enterprise owners, but little additional validation support.
Good	In addition, provides support for capturing review comments (redlining) and comparing them to constructed model.
Strong	In addition, provides support for selected parameter validation checks & identifying differences between the behaviours of a constructed model and the actual enterprise.
Excellent	In addition, provides support for quantifying degree of behavioural validation of a constructed model to the actual enterprise.

10.9.6 T6 – Technical Assessment Criteria – Evaluate and Select

T6.1 Capability to Define Evaluation Metrics & Parameters

None	Cannot define metrics & parameters for assessing added value.
Weak	Can define evaluation parameters of a 'canned' evaluation approach using fixed metrics, but little additional evaluation support.
Good	In addition, provides support for specification and tailoring of evaluation metrics to meet user-specified requirements.
Strong	In addition, provides support for weighting of metrics to meet user-specified preferences.
Excellent	In addition, provides support for tailoring of different sets of metrics with specific target values for each entity to be evaluated.

T6.2 Capability to Evaluate Models

None	Cannot support evaluation of a model for assessing added value.
Weak	Can support evaluation of a model with one 'canned' evaluation approach, but little additional evaluation support.
Good	In addition, provides support for specification and tailoring of evaluation approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one evaluation approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing evaluation approaches in real-time based-upon data and/or conditions within the model (self-evaluating).

T6.3 Capability to Display Evaluation Metrics

None	Cannot support display of evaluation metrics in a model for assessing added value.
Weak	Can support simple display of evaluation metrics in a model with one 'canned' display approach, but little additional evaluation support.
Good	In addition, provides support for specification and tailoring of display approach to meet user-specified requirements.
Strong	In addition, provides support for use and tailoring of more than one display approach with associated capabilities to isolate user from the internal intricacies of the model so as to make the modelling appear 'simple and automatic'.
Excellent	In addition, provides support for automatically and dynamically changing display approaches in real-time based-upon data and/or conditions within the model.

T6.4 Capability to Direct Improvement Based Upon Evaluation Results

None	Cannot direct improvement activities based upon evaluation results.
Weak	Can suggest general improvement opportunities within a model, but little additional evaluation support.
Good	In addition, provides support for identifying specific improvement opportunities within a model.
Strong	In addition, provides support for diagnosing and high-lighting the causes of identified problem areas.
Excellent	In addition, provides support for automatically and dynamically directing optimization activities (refer **T4.8**) in real-time based-upon data and/or conditions within the model (self-improving).

10.9.7 T7 – Technical Assessment Criteria – Implement and Monitor

T7.1 Capability to Use Model Data to Implement Enterprise

None	Cannot implement enterprise from model data.
Weak	Can describe TO-BE enterprise at a high-level to enterprise implementers, but little additional implementation support.
Good	In addition, provides support for specification of automated enterprise & systems requirements in formats helpful to implementers.
Strong	In addition, provides support for automated generation of electronic workflow instructions to prepare for enterprise implementation.
Excellent	In addition, provides support for automated software code generation, configuration commands, etc. to implement/maintain application systems.

T7.2 Capability to Monitor Actual Enterprise for Behaviour Data

None	Cannot support monitoring of enterprise to collect behaviour data.
Weak	Can support recording of opinions of simple high-level behaviour data, but little measurement support.
Good	In addition, provides support for actual measurement of simple high-level parameters (i.e. elapsed time, quantity of late items, etc.) .
Strong	In addition, provides support for automated measuring of more complex lower-level (more task oriented) parameters.
Excellent	In addition, provides support for measuring in real-time complex lower-level parameters of enterprises to be used as input to real-time data preparation.

T7.3 Capability to Prepare Data for Model Behaviour Attribution

None	Cannot support preparation of data for attributing a model for behavioural activation.
Weak	Can support formatting of data, but little additional data preparation support.
Good	In addition, provides support for filtering, reformatting, aggregating and otherwise preparing data to be used as attribute data in a digital model for behavioural activation.
Strong	In addition, provides support for tailoring automated macros for data preparation to be used as attribute data in a model for behavioural activation.
Excellent	In addition, provides support for dynamically preparing data gathered real-time to be used for real-time model attribution.

T7.4 Capability to Attribute Model for Exercising Behaviour

None	Cannot specify and/or update attributes in model to prepare for behavioural activation.
Weak	Can update 'canned' attributes for behavioural activation, but little additional attribution support.
Good	In addition, provides support for specification and tailoring of attributes to meet user-specified requirements.
Strong	In addition, provides support for automatic updating of model attributes with data prepared for modelling.
Excellent	In addition, provides support for dynamically updating attributes in real-time using data prepared in real-time to prepare for real-time behavioural activation.

10.9.8 M – Manageability Assessment Criteria

M.1 Capability to Manage Modelling Capability Requirements

None	Cannot distinguish specific modelling capability requirements.
Weak	Can distinguish specific modelling capability requirements, but little additional capability requirements management support.
Good	In addition, provides a comprehensive enterprise modelling problem space framework to aid in classification of requirements.
Strong	In addition, provides support for identifying varying requirements by source organization and a means for identifying commonalities of requirements across organizations.
Excellent	In addition, provides support for quantifying need for each requirement.

M.2 Capability to Manage Modelling Capability Improvement

None	Cannot identify promising areas where modelling capability can be improved.
Weak	Can identify promising improvement areas, but little additional capability improvement support.
Good	In addition, provides support for recording and tracking suggested improvements to these areas.
Strong	In addition, provides support for identifying degrees of improved capability and cost of usage for each technique using the common framework.
Excellent	In addition, provides support for quantifying improved capability and/or cost of usage of a technique and evaluating the improved technique for value-added over unimproved techniques.

M.3 Capability to Manage Modelling Standardization

None	Cannot support definition of any modelling standards
Weak	Can support definition of a modelling standard, but little additional standards support.
Good	In addition, provides support for identification of modelling standard overlaps and gaps according to a common framework.
Strong	In addition, provides support for identifying degrees of capability and cost of using each standard technique using the common framework.
Excellent	In addition, provides support for quantifying capability and cost (i.e. value) of a standard technique

M.4 Capability to Manage Model Consistency

None	Cannot support comparison of working models to standards.
Weak	Can support comparison of working models to standards, but little additional model consistency support.
Good	In addition, provides support for identification of modelling standard violations according to a common framework.
Strong	In addition, provides support for identifying degrees of capability and cost of violations using the common framework.
Excellent	In addition, provides support for quantifying capability and cost (i.e. loss of value) of standards violations and preventing violation where the loss of value exceeds specified parameters.

M.5 Capability to Manage Configuration Control of Models

None	Cannot support configuration control of models.
Weak	Can support identification of versions, but little additional configuration control support.
Good	In addition, provides support for identification of alternatives within versions and control and audit trail of versions and alternatives.
Strong	In addition, provides support for identifying and resolving differences and conflicts between alternatives and versions of a model and identifying degree of integrity possessed by a current version.
Excellent	In addition, provides support for quantifying integrity of current versions of models using a common framework.

10.9.9 U – Useability Assessment Criteria

U.1 Capability to Be Used Anywhere In Company (Commonality)

None	Not currently used anywhere in company.
Weak	Currently used in a couple of locations within the company.
Good	Currently used in scattered locations within the company.
Strong	Currently in common use throughout the company.
Excellent	Currently a company standard.

U.2 Capability to Be Learned (Learning Curve)

None	Requires a year of experience with the tool before meaningful models could be built.
Weak	Requires several months of experience with the tool before meaningful models could be built.
Good	Requires less than a month of experience with the tool before meaningful models could be built.
Strong	Requires less than a week of experience with the tool before meaningful models could be built.
Excellent	Requires less than a day of experience with the tool before meaningful models could be built.

U.3 Capability to Be Helpful to the Modeller (Help)

None	Tool contains no help aids and documentation is non-existent.
Weak	Tool contains a few help aids and/or documentation is poor.
Good	Tool contains general help aids and documentation clearly describes all basic functions.
Strong	Tool contains specific help aids and documentation clearly describes all functions.
Excellent	In addition, tool contains context-sensitive help aids.

U.4 Capability to Accomplish Work (Power)

None	Requires a year for an experienced modeller to build a small model (100 entities and relationships).
Weak	Requires several months for an experienced modeller to build a small model (100 entities and relationships).
Good	Requires less than a month for an experienced modeller to build a small model (100 entities and relationships).
Strong	Requires less than a week for an experienced modeller to build a small model (100 entities and relationships).
Excellent	Requires less than a day for an experienced modeller to build a small model (100 entities and relationships).

10.9.10 D – Deployability Assessment Criteria

D.1 Capability to Work on Models on Any Platform (Accessibility)

None	Cannot work on models anywhere.
Weak	Models can only be worked-on on the same platforms on which they are created.
Good	Models are compatible at the source level on multi-vendor platforms of the same operating system type.
Strong	Models are compatible at the binary level on multi-vendor platforms of the same operating system type.
Excellent	Models are compatible at the binary level on multi-vendor platforms of different operating system types.

D.2 Capability to Build Models of Any Size (Scalability)

None	Cannot build models of any size.
Weak	Small models can be built (1 to 1000 logically consistent entities and/or relationships).
Good	In addition, medium models can be built (1000 to 100,000 logically consistent entities and/or relationships).
Strong	In addition, large models can be built (100,000 to 10,000,000 logically consistent entities and/or relationships).
Excellent	In addition, huge models can be built (¿10,000,000 logically consistent entities and/or relationships).

D.3 Capability to Extend Modelling Capabilities (Extensibility)

None	Cannot extend modelling capabilities at all.
Weak	Can adjust entity and/or relationship type attributes.
Good	In addition, can add new entity and/or relationship types.
Strong	In addition, can adjust functionality of menu selections without programming.
Excellent	In addition, can add and integrate any new functionality as needed.

...

D.4 Capability to Tailor Tool to Meet Needs (Flexibility)

None	No choice in tool versions(one tool, one price)
Weak	Can choose tool versions to meet modelling size needs (i.e. small models, large models, etc.)
Good	Can choose tool versions to meet modelling task needs (i.e. development, modelling, etc.)
Strong	Can choose tool versions to meet both modelling size and task needs.
Excellent	Can tailor tool capabilities (and cost) to meet specific customer needs.

D.5 Capability to Model with Reliability (Maturity)

None	Tool has not been in production use.
Weak	Tool has been in production use less than a year.
Good	Tool has been in production use in scores of locations for at least two years.
Strong	Tool has been in production use in hundreds of locations for at least three years.
Excellent	Tool has been in production use in thousands of locations for at least three years.

D.6 Capability to Adhere to Computing Architectural Standards

None	Tool does not adhere to any computing architectural standards.
Weak	Tool is partially compliant to applicable computing architectural standards.
Good	Tool is mostly compliant to applicable computing architectural standards..
Strong	Tool is totally compliant to applicable computing architectural standards.
Excellent	In addition, tool supports significant additional international standards.

D.7 Capability to Train People to Use Tool (Training)

None	No training or training aids available.
Weak	Minimal training or training aids available.
Good	Strong, affordable training is available.
Strong	In addition, strong training aids (training manual, videos, etc.) are available.
Excellent	In addition, strong training aids are built into the tool and available whenever needed by modeller (self-training).

D.8 Capability to Acquire Assistance If Needed (Support)

None	No support available.
Weak	Minimal support available (unknowledgeable and/or poor response).
Good	Strong, affordable support is available (knowledgeable and reasonably responsive).
Strong	In addition, knowledgeable locally available support is available.
Excellent	In addition, knowledgeable on-site support is available.

D.9 Capability to Afford Tools (Affordability)

None	Per-seat cost (fully deployed) is greater than $25K.
Weak	Per-seat cost (fully deployed) ranges from $10K to $25K.
Good	Per-seat cost (fully deployed) ranges from $5K to $10K.
Strong	Per-seat cost (fully deployed) ranges from $1K to $5K.
Excellent	Per-seat cost (fully deployed) is less than $1K.

Enterprise Modelling Technology Capability Assessment Chart

Technology: ________________________________

Capability to...	None	Weak	Good	Strong	Excellent
T1.1 Model Enterprise Entities					
T1.2 Model Entity Relationships					
T1.3 Capture Enterprise Data From All Necessary Sources					
T1.4 Model All Stages of Enterprise Improvement Life-cycle					
T1.5 Model Completeness of Enterprise Knowledge					
T1.6 Model Predictability of Enterprise Knowledge					
T1.7 Model Precision of Enterprise Knowledge					
T1.8 Model Certainty of Enterprise Knowledge					
T1.9 Model Entity Locations					
T1.10 Model Entity States					
T1.11 Model Entity Interactions					
T1.12 Model Entity Temporality					
T1.13 Model Entity Capability					
T1.14 Model Entity Cost					
T1.15 Model Enterprise Risk					
T1.16 Perform Abstraction of Model					
T1.17 Perform Refinement of Model					
T2.1 Maintain Data Consistency					
T2.2 Manage Model Partitioning					
T2.3 Manage Model Interfacing					
T2.4 Manage Conflict Resolution Between Models					
T2.5 Manage Model Integration					
T3.1 Tailor Representations Used in Model					
T3.2 Control Manipulation of Entities in Model					
T3.3 Control Printed Output of Model					
T3.4 Tailor Visual Controls of Model Behaviour					
T3.5 Tailor Visualization of Model Behaviour					
T4.1 Construct Analysis Model					
T4.2 Analyse Model Structure (Static Analysis)					
T4.3 Analyse Model Behaviour (Dynamic Analysis)					
T4.4 Perform Discovery in Models					
T4.5 Configure Models					
T4.6 Design Models					
T4.7 Adjust Models					
T4.8 Optimise Models					

(continues on next page) 1/2

Fig. 10.12. Capability Assessment Chart (© NCR 2001, used with permission)

Enterprise Modelling Technology Capability Assessment Chart

Technology: ________________________________

Capability to...	None	Weak	Good	Strong	Excellent
T5.1 Assess the Appropriateness of Formalism					
T5.2 Assess the Range of Applicability of Formalism					
T5.3 Verify Model Construction					
T5.4 Validate Model Structure					
T5.5 Validate Model Behaviour					
T6.1 Define Evaluation Metrics \& Parameters					
T6.2 Evaluate Models					
T6.3 Display Evaluation Metrics					
T6.4 Direct Improvement Based Upon Evaluation Results					
T7.1 Use Model Data to Implement Enterprise					
T7.2 Monitor Actual Enterprise for Behaviour Data					
T7.3 Prepare Data for Model Behaviour Attribution					
T7.4 Attribute Model for Exercising Behaviour					
M.1 Manage Modelling Capability Requirements					
M.2 Manage Modelling Capability Improvement					
M.3 Manage Modelling Standardization					
M.4 Manage Model Consistency					
M.5 Manage Configuration Control of Models					
U.1 Be Used Anywhere In Company (Commonality)					
U.2 Be Learned (Learning Curve)					
U.3 Be Helpful to the Modeller (Help)					
U.4 Accomplish Work (Power)					
D.1 Work on Models on Any Platform (Accessibility)					
D.2 Build Models of Any Size (Scalability)					
D.3 Extend Modelling Capabilities (Extensibility)					
D.4 Tailor Tool to Meet Needs (Flexibility)					
D.5 Model with Reliability (Maturity)					
D.6 Adhere to Computing Architectural Standards					
D.7 Train People to Use Tool (Training)					
D.8 Acquire Assistance If Needed (Support)					
D.9 Afford Tools (Affordability)					

2/2

Fig. 10.13. Capability Assessment Chart (continued)(© NCR 2001, used with permission)

11

MODELLING FUNCTION AND INFORMATION

Peter Bernus

Griffith University (Australia)

11.1 Introduction

One of the most important tasks in the requirements life-cycle phase is to determine the functions that the given enterprise entity must have and how these functions interact – i.e., through exchanging information and material. For this purpose, one has to determine what the user (business owner) would like to include in the specified entity. The result may be called the *user requirements specification* of functions and data. These requirements may be captured by analysing the *business processes* that the enterprise intends to implement. The task is usually performed by business analysts, who:

- interview management and other technical personnel to capture the functions to be included in the new business processes,
- investigate various documents (such as quality manuals, descriptions of existing procedures and processes) in order to get a complete picture of what is required.

Once the functional and information requirements are expressed by analysts, the resulting documents must be validated by business users, to *verify* that the analysis captured the requirements correctly.
The identified business processes either manage material and information flow in order to satisfy customer's needs (i.e., mission fulfilment – such as product design, manufacture or transport of goods, service provision), or co-ordinate and control the present or future operation.
Mission fulfilment and management processes are different in nature, and their specifications call for different modelling languages and tools. This is because management processes are usually creative and non-procedural, and characteristics that need to be designed into the management system are different

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

from those to be designed into the mission fulfilment system[1]. For example, many management processes are policy driven rather then procedure driven. Mission fulfilment needs a mixture of material and information processing and important considerations include throughput, processing speed, need for transport and storage, and a number of other logistics functions. This Chapter will concentrate on the characteristic modelling tasks of mission fulfilment, including the modelling of procedural / mechanistic, as well as creative functions.
After the validation of user requirements, analysts translate these documents into complete specifications that are:

- detailed enough to be able to be used for the preliminary design of the enterprise entity;
- compliant with good design principles (so as to create a specification that can be implemented, i.e. they are feasible and affordable in light of the possible technologies that may be used for implementation);
- robust (in light of likely future changes);
- generally of good quality (complete and compliant with quality standards, such as ISO 9000:2000, and other relevant industry specific quality and safety standards), compliant with other standards (such as industry-wide or international standards regarding processes and data exchange);, and
- potentially related to previous design efforts (so as to be able to reuse previous experience and knowledge that may be available in the form of reference models that capture industry best practice or other known good solutions).

The result of this work is a *systems requirements specification.* Therefore, this Chapter is about:

- capturing the functional and information requirements as ascertained and validated by the user (business owner) and
- developing a systems requirements specification of functions and data.

Methodologies often concentrate on the functions to be performed and information to be managed by automated means, such as computer software and hardware, or control equipment (with embedded software). The human function and information is either neglected or is analysed only to the extent that allows the specification of the user interface of any automated equipment to be designed. The problem with this approach is that it prevents the systems architect (who performs the preliminary or 'architectural' design) from making decisions about the level of automation that is desirable for the enterprise[2].

[1] note that some, but certainly not all, mission fulfilment processes share some of the characteristics of management processes, such as in product research, design and development

[2] since that level is 'set' in an implicit manner, rather by explicit specifications, and thus may not be readily altered

Therefore, the resulting systems requirements specification cannot be easily reused if the desired level of automation changes in the future.
The preferred development of a requirements specification for function and information should be done – to the extent possible – irrespective of the means by which these will be implemented. It is then the preliminary design activity that decides on the level of automation of processes, as well as what information is stored in computers and control equipment, as opposed to being stored by other means (in people's heads, on paper, etc).
This (quite common) notion of separation of the function from its means of implementation may lead to non-implementable specifications, unless exercised with caution. However, it is the task of the user requirements specification to pin down any constraints, where there is only a very limited (or unique) choice in the way the function is implemented, thus limiting the extent to which the systems requirements specification may exercise creativity. E.g., if a business process is designed around some significant asset (machinery, infrastructure), then there is no room to change the interfaces of functions that interact with it. Even so, it is customary to *wrap* such mandatory components of a system into a more generic interface, so that future changes may be easily carried out when such a component is replaced.

11.2 Modelling the Function of the Enterprise Entity

There are several ways to capture the functions that the enterprise must be able to perform in order to fulfil its mission. Some functions directly contribute to the (information and material) transformation that eventually delivers goods or services to the customer, while some others are *support processes* that are necessary for the main business processes to be performed.
It is rarely the case that all of the above processes need to be specified in detail in any single-change endeavour. Thus, the realm of mission fulfilment activities in the enterprise may be grouped into *domains.* Some of these are to be specified in detail and some others (that are intended not to change) need to be specified only to the extent of their interfaces, so as to sufficiently constrain the specification of new (or 're-engineered') processes. These interfaces involve material and data flow, and events. Therefore, the interfaces of business processes that are outside the modelled domains will still have to be specified.
The functional models are used

- to design new or changed business processes, or
- to design how to integrate existing business processes (in which the existing information or material flow is too slow, or the performance characteristics of the existing processes are not satisfactory).

This second case often arises if a business process cuts across enterprise boundaries, and also if the control and feedback information between the mission

fulfilment processes and the management and control processes is not satisfactory.

In talking about modelling the functions of the enterprise it is necessary to define the concept of *function*[3] (as opposed to process, procedure or activity).

Definition: *function*

A function is a *transformation* of inputs (material and/or information) into outputs. To specify this transformation means to describe what the output(s) are going to be for each possible (admissible) combination of inputs.

The specification of a function must describe the:

- Inputs
- Transformation
- Outputs

Thus, a function is like a black box described by its interfaces and a transformation. Sometimes this transformation can be expressed as a mathematical function – in which case the function can be implemented by a procedure (algorithm) – but often it is provided in another form.

The necessary level of detail for a functional specification is determined by the intended audience. If there are *guarantees* that the specification will not be mis-interpreted in any way by the intended audience, then the specification may be deemed complete.

11.2.1 Functional Decomposition

There are several ways to define a function apart from a mathematical transformation. In modelling business functions, the ability to use a mathematical transformation for specification is limited and suitable only for some specifications.

One usual approach is to describe the function as a result that emerges from the *interaction* of some more elementary (sub)functions. This is called a *functional decomposition* and the resulting model a *functional model.* Using functional decomposition, the set of elementary functions, as well as the interaction among these functions must be made clear. A notable enterprise modelling language to describe this functional decomposition is IDEF0 (Menzel and Mayer , 1998; IEEE , 1998a). Figure 11.1 shows a typical business function and its decomposition into more elementary functions. In the figure boxes represent component functions, while arrows represent inputs and outputs.

Functional decomposition may be performed through multiple levels, where sub-functions are further decomposed into even more elementary functions, etc. Keeping in mind that these functional models are predominantly used for

[3] some architecture frameworks (e.g. the Purdue Enterprise Reference Architecture - PERA) use the word 'task' instead of 'function' but with the same meaning

the communication of the requirements specification between users and analysts, as well as between the analysts and systems architects, the criterion of completeness of such functional decompositions is that the person (or group) who produced the functional specification should successfully communicate any intended meaning of these documents to the person who needs to use this specification. For this reason, functional specifications should be accompanied with *explanations* and *references* to objects and processes in the environment that all of the involved parties are familiar with.
The meaning of a functional specification relies on the meaning of the most elementary functions reached in the functional decomposition. Such an elementary function is called an *activity.*
There are several ways to specify the meaning of elementary functions:

- The elementary function may be sufficiently specified by the usual meaning carried by the name of the function. This is possibly enriched by textual explanations taking into consideration the context of the specified inputs, outputs and the function of which the given elementary function is a part. Explanation is also needed to set the context of the specification document – i.e. who prepares the specification and who is the intended audience;
- The elementary function is defined as a mathematical function of inputs to outputs;
- An elementary function may be further specified by describing a procedure, i.e. instead of further functional decomposition the function is specified by an algorithm (refer Section 11.2.2 below);
- The elementary function is specified by identifying a resource that has such a function, meaning that given the right controls, the resource is capable of performing the desired transformation.

In practice, it is necessary to also specify the capacities and capabilities required by the function, so as to ascertain that a later selected resource could indeed be applied to perform the given function (refer Chapter 13 for the discussion of resource modelling). E.g., for each function in Fig. 11.1 the expected rate, volume and number of inputs to be processed, and the expected time to perform the transformation would need to be specified.

11.2.2 Procedural Decomposition

Another way to describe a function is by specifying a *procedure* that, when performed, results the desired transformation. A generic definition of a procedure is given below (this definition is quite abstract, hence a concrete example is given after the generic definition). The reason for this is to have a common definition that is applicable to all modelling languages that are procedural in nature (such as the CIMOSA process model (Vernadat , 1996; CIMOSA , 1996), IDEF3 (Menzel and Mayer , 1998; Mayer et al. , 1993), ARIS Event Driven Process Chain (Scheer , 1998), Petri Nets (Petri , 1962;

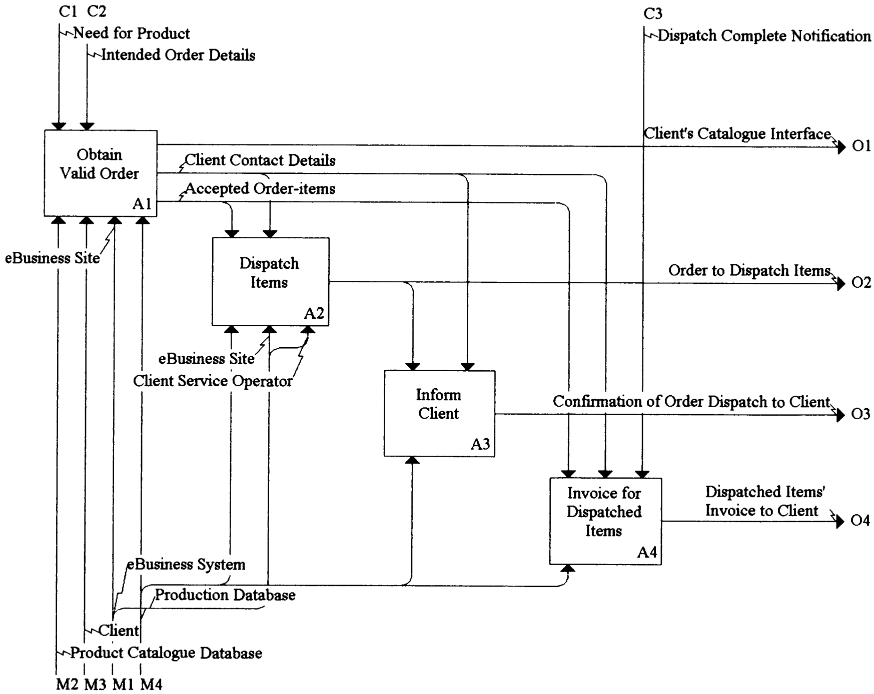


Fig. 11.1. A typical functional model (represented in IDEF0)

Peterson , 1981; Murata , 1989; Jensen , 1997a,b), Process algebra (Hoare , 1985; Hennessy , 1988; Baeten and Weijland , 1990) and LOTOS[4] (van Eijk et al. , 1989; ISO , 1989), Unified Modelling Language (UML) collaboration diagrams, UML sequence diagrams (Rumbaugh et al. (1998) and ISO/IEC (2001)), etc).

[4] a language based on process algebra

Definition: *procedure*
A procedure is a special type of function defined by its *behaviour*, which in turn may be defined by specifying *units of behaviour*[5] *(UoB)* and *precedence relationships* between these. An UoB may then be defined again either as a function or a procedure.
The main characteristic property of a procedure is that its description is *executable.* Succession relationships are expressed as an acyclic directed graph, where nodes are units of behaviour and arcs are succession relationships. (While it is customary to use cyclic graphs for the specification of a procedure, feedbacks should be interpreted as shorthands for repeating the same graph again from the point of feedback.) A procedure can be *invoked*, and as a result it gets *executed.* If a procedure is executed then one of its UoBs is activated and when this UoB is completed, one or more other UoBs are activated according to the defined precedence relationships. A sample procedure is shown in Fig. 11.2, where large boxes stand for UoBs and directed arrows represent precedence relationships.
Precedence relationships may be simple (represented by an arrow pointing from a UoB 'A' to another UoB 'B', meaning that 'A' must finish before 'B' starts), while other precedence relationships are more elaborate and are called 'junctions'. These 'junctions' are represented using special graphical symbols (the small boxes in Fig. 11.2).
Typical junctions with one input arc and two or more output arcs are:

- the 'AND' junction: this junction, when the UoB at its input has finished activates all UoBs on its output;
- the 'OR' junction, which non-deterministically[6] activates one or more UoBs on its output;
- the 'XOR' junction that non-deterministically activates exactly one UoB on its output.

Typical junctions with two or more input arcs and one output arc are:

- the 'AND' junction that activates its output UoB when all of its input UoBs have finished ;
- the 'OR' junction, which activates the UoB on its output as soon as at least one of its input UoBs have finished. Synchronous versions of these junctions also exist (see Menzel and Mayer (1998)).

Note that an arc in the directed graph may be interpreted as the final point in time of an event (where an event is one activation of a unit of behaviour, starting at a given time point and finishing at a time point).
Practically, a procedure is a sequence of steps that can be executed, sometimes iterated, where each step is either a (sub) procedure itself, or an activity. It

[5] the term 'unit of behaviour' is borrowed from the IDEF3 language
[6] the UoB activated depends on execution conditions that may be attached as a textual or formal description of the junction itself.

is possible to describe the time, cost and necessary capacity associated with each step – therefore, depending on the reason for modelling the procedure, the cost, time, capacity, etc. of the procedure can be predicted.

To completely specify a procedure all UoBs comprising it must be defined. Since UoBs are procedures and procedures are special functions, the same choices are true for specifying UoBs as listed in Section 11.2.1. Having these choices is important, because one might want to specify a procedure, where the UoBs are creative functions, in which case the higher-level procedure links functions that can not be described themselves as procedures.

A typical procedural model (also called a *process model*) is shown in Fig. 11.2[7] using the IDEF3 language. Even though there are several differences between the languages listed in the beginning of this section, all share the basic property of being procedural.

Note that for the practical modelling task junctions need to be associated with *states* of objects (i.e., information and/or material) specifying the state of objects used and produced by units of behaviour. E.g. a UoB may have a function that has the same input and output object, but in different states.

If many objects are involved in the procedure and their states are changed by the procedure, then it soon becomes difficult to see whether the described procedure will indeed perform the desired state transitions, and whether all desired state transitions are actually available. For this reason procedural modelling languages may be extended with a language that can represent how units of behaviour (i.e. 'steps' of the process) change object states. E.g., the IDEF3 language includes Object State Transition diagrams and UML includes Harel State Charts – with the advantage of the latter that object states can be clustered / structured (Desharnais et al. , 1998).

Different enterprise modelling languages and tools that support them take different approaches to the definition of object states and attributes of these states. Coloured Petri Nets for example allow the association of *attributes* with objects that flow between units of behaviour. The CIMOSA-based First-Step tool allows the association of a state with objects that flow between units of behaviour, and ERWin / BPWin allows user defined attributes for all modelling constructs. Configurable tools (e.g. FrameWork, System Architect) practically allow the same (these last three modelling tools give the user some limited ability to extend the meta-schema of the underlying languages, although the meaning attached to such an extension is not known to the tool)[8]. This extendibility is useful because it allows models to represent user-defined properties. However, the problem is that end users need standards for these

[7] The original figure has been extended with proper IDEF3 junctions. Note that this procedure contains functions (UoBs) that are themselves usually non-procedural – therefore further definition of these may have to be given using a functional model, which in turn may contain lower level functions that can again be defined procedurally, etc.

[8] for a description of the ARIS, FirstStep, System Architect and FrameWork modelling tools refer to Chapter 3

extensions, otherwise the exchange of such extended models is limited to the project or enterprise in question.
CIMOSA defines many useful additional attributes, and it is expected that the much awaited UEML[9] specification will define a sufficiently complete set of attributes that do not need user defined extensions, except in very unusual circumstances.

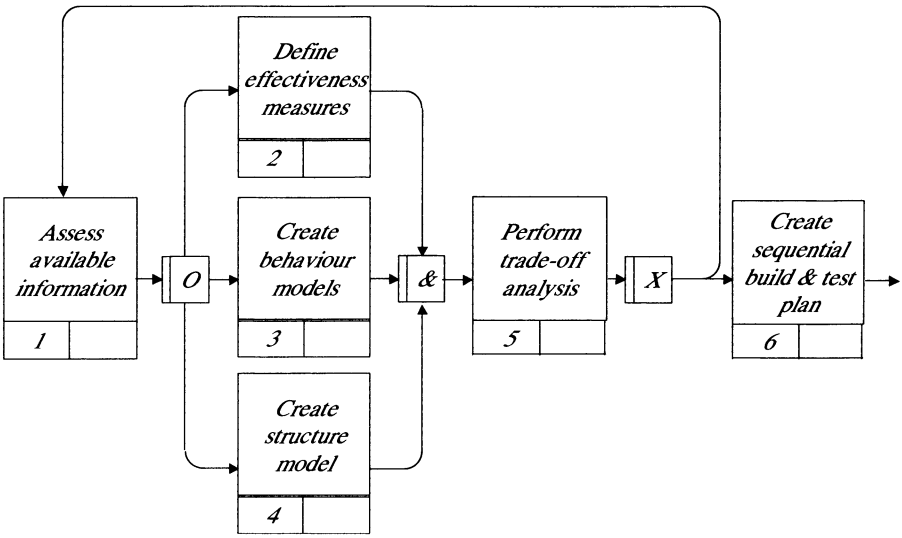


Fig. 11.2. A typical procedural model describing the high level technical processes of systems engineering (represented in IDEF3) (adopted from Oliver et al. (1997)

11.2.3 Combined (Functional and Procedural) Decomposition of Functions

A practical model of business processes needs to combine functional and procedural specifications. E.g. on the highest level of describing a business process, such as New Product Development (NPD), the enterprise might want to enforce a procedure, where certain steps must precede others (introducing milestones, or 'decision gateways', for example – refer Chapter 19). Clearly, NPD includes functions that cannot be expressed as pure procedures, therefore they can only be described using a functional decomposition. In this

[9] UEML: Unified Enterprise Modelling Language (at the time of this writing being developed by the UEML Project supported by the European Community). Refer to the website at http://www.UEML.org

functional decomposition, however, some functions may be procedural and some non-procedural. Thus, the model of NPD would be best expressed as a mixture of functional decompositions and procedures.
Unfortunately, enterprise modelling tools are not very strong in this respect. E.g. FirstStep and other process modelling tools insist on a procedural decomposition of all processes. ARIS allows a functional decomposition (but without the information and material flow specified between the sub-functions) and each sub-function can be described as a procedure (using Event Driven Process Chains). IDEF0 and IDEF3 tools allow the mixing of functional and procedural decompositions, but with little help for describing additional attributes, such as capacities and necessary capabilities (where FirstStep and ARIS are strong).
As a result, at the present time modelling of the business processes (functionally or procedurally) needs to be accompanied by textual descriptions to fill the gap between the expressiveness of the underlying languages and the needs of the business process modeller. It is expected, however, that tool developers will close this gap, as soon as the necessary standards are developed.

11.2.4 To What Detail is Information Modelled in a Functional Specification?

A functional decomposition effectively constructs the meaning of a function through a *composition* of elementary functions. The IDEF0 language uses a graphical notation, where boxes stand for functions and arrows stand for the inputs and outputs – with the flow of arrows representing the composition.
Theoretically, the input and output specifications of a functional decomposition could also be used as the specification of the information aspect of the business process – all that would be necessary is to add sufficient detail to the description of these input and output objects. However, in practice it is better to separate the exact specification of functions from the *exact* specification of information. This is because it is common that many functions refer to the same kind of information object (albeit to different parts of it or to different states). To ensure that a functional decomposition does not become redundant, instead of repeating a detailed descriptions of inputs and outputs (information objects) many times over, a functional specification can simply *name* the information objects exchanged between functions (possibly referring to their state) and only refer to a *separate* specification of the structure and content of the information objects (refer Section 11.3). Another reason for a separate specification of information objects is that the relations between objects cannot be graphically represented in a functional decomposition.

11.3 Modelling Information

The modelling of information is the most advanced field of enterprise modelling. The reader may have noticed that in this Chapter the words 'data' and

'information' were used (up to this point) almost interchangeably. In fact they are not the same. It is commonly known that it is data that one can capture, while information is the *newsworthiness* of data. I.e. humans or machines, by interpreting data, can make decisions, and if the interpretation is correct and consistent across users, then data exchange is deemed to have served information exchange. Therefore two things are required to adequately specify the *information flow* within and among business processes:

- the structure of data must be specified (data modelling) – in form of data schemata;
- the conditions of correct interpretation must be established.

Data modelling is preoccupied with the first of these requirements. The correct interpretation is then ensured by controlling the context in which data are exchanged. The functional requirements specification (Section 11.2) can be used to define this context.

Figure 11.3 shows a high level methodology to carry out functional specification and data modelling in a quasi-parallel fashion. The methodology consists of a progression from an initial functional specification (to establish the context and scope of the data that must be modelled), through the development of a model of data, and the completion of the functional specification as well as the specification of data. In practice, the process needs iteration and feedback between these activities.

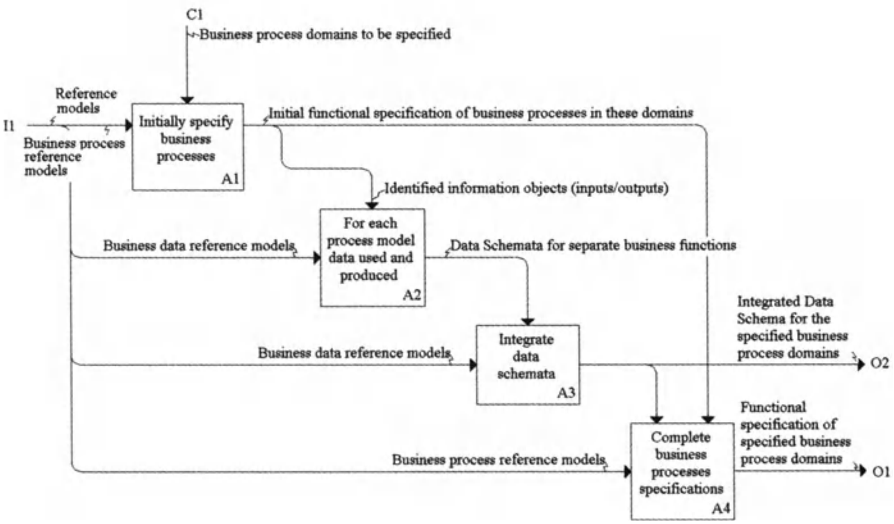


Fig. 11.3. High-level methodology for specifying functions and data [10]

[10] feedbacks / iterations not shown for simplicity.

Also, the figure reminds the reader to the fact that modelling does not start from scratch: reference models (either standard or gathered from previous experience) can, and should be used.
There are several modelling languages that can be used to develop data schemata on the *requirements* level. The most commonly used ones are the Entity Relationship (ER) data model[11] (Chen , 1976) – in fact the Extended ER Data Model is used today citepBat92,Elm00 – and IDEF1X (NIST , 1993; Menzel and Mayer , 1998; IEEE , 1998b) that may be considered a standard notation for the ER data model.
Furthermore, various forms of the object oriented data model can be used for data modelling, although their power to express various constraints varies largely. UML class diagrams are the simplest (for a comparison see Halpin (1999)), while EXPRESS (ISO , 1992) and Object Role Modelling (ORM) (Halpin , 1998) have standard notation for a variety of business constraints. Less expressive languages relegate such constraints to textual description.
An enterprise model representing data is called a Data Schema[12]. Figure 11.4 shows a typical Entity Relationship schema (using the Extended ER data model)[13].
Note that First Order Logic (FOL) and its various notations, such as Z (Spivey , 1989), Object-Z (Smith , 2000), KIF (Genesereth and Fikes , 1992) and Conceptual Graphs (Sowa , 1984), are also candidate languages that can be used for data modelling. For a review of data (information) modelling languages see Mylopoulos (1998).
While the expressive power of the above languages is slightly different, the choice of the language should not be solely based on the power of the particular language.

---

[11] A 'Data Model' is the formal specification of a data modelling language. The language can be specified as a metaschema. A data model usually identifies entities (objects), attributes (properties of entities), attribute domains (values that attributes can have), relations among entities (associations), key constraints (ensuring that each instance of an entity or object is uniquely identifiable), cardinalities of relations (how many relationship instances a given entity instance can participate in), as well acknowledges the existence of semantic integrity constraints (constraints that can not be expressed using the basic notation of the language and therefore must be added to the Data Schema using some other notation – such as text, or a notational variant of First Order Logic)

[12] Literature prefers to use the term Database Schema, but as pointed out in the beginning of this chapter, some data represented may not be stored in a database. Alternatively, the Data Schema may be called an *Information Schema*, with the understanding that the schema should be used in well-understood and specified contexts to ensure correct interpretation.

[13] Unfortunately, the Entity Relationship Data Model does not have an international standard graphical notation, and various case tool vendors use slightly different graphical conventions.

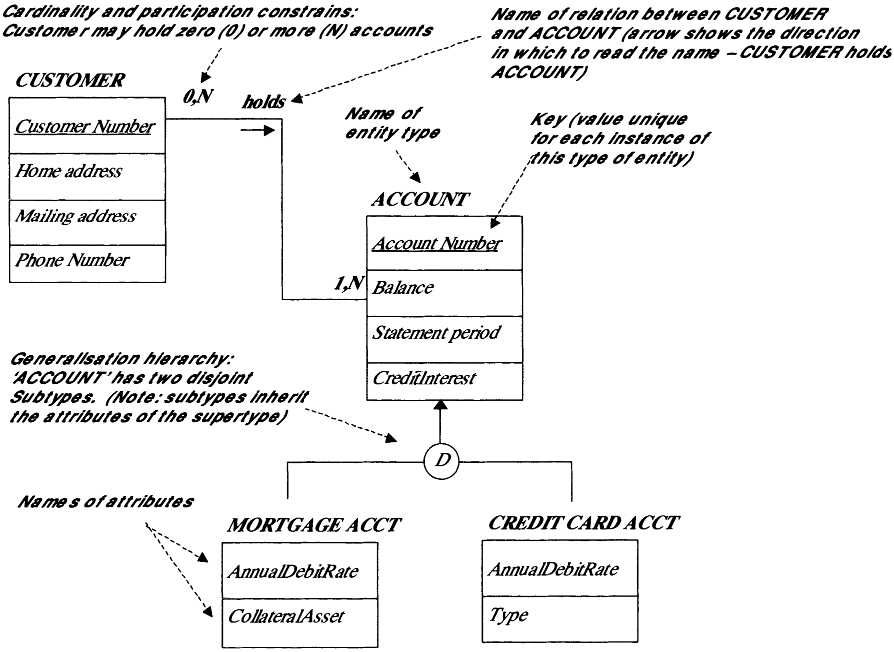


Fig. 11.4. A typical Data Schema expressed using the Extended Entity Relationship data model[14]

The local culture, familiarity with the selected language, the availability of a modelling tool that can integrate the functional and data modelling aspects are usually important additional factors to consider (refer Chapter 10).

11.4 Reference Models for Function and Data Modelling

For achieving an integrated information flow within the enterprise's business processes and at the same time ensuring that these processes communicate with the external environment (such as customers and suppliers), the development of data schemata must take into account international or industry standards. Disregarding such standards poses two dangers:

- the enterprise may develop a data schema that is not implementable using commercial off-the-shelf tools, and therefore in-house development and maintenance costs are incurred, alongside with other risks that such development carries;
- the enterprise is able to implement its internal information system using commercially available modules, but the interoperability of its information system with systems of other enterprises is not ensured.

[14] the explanation of the particular notation used is shown in italics

Both of these dangers are very serious and no enterprise should take uninformed decision in this regard.
Notable data exchange standards that are international or de-facto accepted by industry are, for example:
Product data exchange standards developed by the STEP community of the International Standards Organisation (ISO) (ISO TC184/SC4/WG1). These standards are under continuous development and their aim is to define how product descriptions (such as specifications and designs) can be represented in a common way in various branches of industry. The automotive, aerospace and building industries have separate STEP standards for this purpose, and Computer-aided Design systems vendors implement STEP interfaces through which product data can be imported or exported between various systems and used in supply chains. The STEP standards include the definition of the data exchange context (intended use) through a functional specification of the processes (usually in IDEF0) that need product data exchange. The actual data exchange standards are then specified in the EXPRESS data modelling language, which is based on a variant of the object oriented data model.
The Electronic Data Interchange (EDI) community has developed a number of Data Schemata describing the structure and content of data in usual business transactions, such as purchasing and electronic payment. However, the EDI community has not been extremely successful in proliferating the use of its standards, due to the implementation cost associated with special networks and support products, which created an entry barrier for many small and medium enterprises. The Data Schemata defined by EDI are mostly independent of implementation tools. A recent development is the attempt to map EDI schemata on the detailed design level onto XML (XML , 2000; XML, 2001; Harold , 2001) syntax. This will allow a cheaper and more widely available technology platform to be used for electronic data interchange [15].
Separately from the EDI community, several vendors and vendor communities have developed (and continue to do so) electronic business data interchange standards (e.g. Rosettanet[16]). Both of these efforts recognised the need for the definition of the business process context together with the data schemata. EDI and Rosettanet assume that the business processes involved in this interchange are procedural. Consequently, as opposed to the STEP standards, the involved business processes[17] are described in a procedural modelling language (such as UML collaboration diagrams), and the data schemata are expressed as UML class diagrams as part of these standards.

[15] since the EDI standards have been around for some time, it is expected that new information requirements will extend existing EDI data schemata and their XML mappings
[16] at the time of writing, http://www.rosettanet.org
[17] called PIPs (Partner Interface Processes)

An extensive set of reference models including both functions and data is the set of various Communication Protocols and Application Programming Interfaces.

Communication protocols are usually defined using a procedural modelling language, and the procedure's description is extended with some form of state transition diagram, because communication protocols are usually not stateless.

Application programming interfaces (APIs) are usually defined by specifying the function of each method that a given application object offers (e.g. mathematically, with pre- and postconditions), also defining the structure of inputs and of outputs produced as a response. Most application programming interfaces are either stateless, or involve only simple state transitions.

An important type of reference models in data modelling is the Corporate Data Model (CDM). CDMs define the common structure of many related data schemata. Such CDMs do not represent the structure of a particular set of databases[18] (or information domain) in a particular enterprise. Instead, these schemata describe a reusable *pattern* (Hay , 1996) that can be specialised and used in the design of many databases or other data representations. The advantage of such CDMs is that through their use particular / individual database schemata are easier to integrate, because their design considers:

- the need to model the structure of data (due to user needs), and
- the requirement to be *compatible* with similar but distinct other data schemata in the same business domain.

This second requirement is important, because the same information can in fact be represented in a number of ways using data modelling languages. Information exchange between representations that do not take this requirement into account are hard to achieve (e.g. it is difficult to write a database transaction that needs to access the same type of information that is stored in one database using one structure and in another database using a different structure). Examples of such corporate database schemata that allow the interoperation of multiple different implementations are the SIZ[19] data model of the German Savings Banks Organisation (GSBO)(Krahl and Kittlaus , 1998) and the IBM Insurance Application Architecture (Dick and Huschens , 1998). Many CDMs have been developed in industry, especially by large corporations.

[18] The name 'Corporate Data Model' is also used both commercially and in the research literature to mean a single large (and monolithic) database schema incorporating all descriptions of all data of all strategically significant application programs in the company. This might be an appealing concept, but very difficult to implement, unless the scoping of such CDM development project is done with extreme care. In this Chapter, the term CDM is used to mean patterns, or partial models, of corporate data that help companies develop integrated or *integrateable* data schemata

[19] Informatikzentrum der Sparkassenorganisation (the Computer Science Centre of the German Savings Banks Organisation)

In recent years, the concept of design pattern[20] has been used in other areas as well, as popularised by the developers of the UML language (e.g. Larman (1998)). E.g., similarly to patterns in data modelling, design patterns can be developed for function and behaviour models, incorporating programming best practice.

11.5 Conclusion

This Chapter has given a brief overview of function and information / data modelling, without being an introduction to the details of data and function modelling languages. The reader is referred to the Handbook on Architectures of Information Systems (in the same Handbook series) for the detailed descriptions of many of the mentioned modelling languages.

References

Baeten, J.C.M. and Weijland, W.P. (1990) Process Algebra, Cambridge University Press.

Batini, C., Lenzerini, M. and Navathe, S. (1992) Database Design: An Entity-Relationship Approach, Benjamin Cummings.

Chen, P. (1976) The Entity-Relationship Model: Towards a Unified View of Data, ACM Transactions on Database Systems 1(1).

CIMOSA (1996) CIMOSA Formal Reference Base, Version 3.2. Böblingen, Germany : CIMOSA Association e.V.

Desharnais, J., Frappier, M., Mili, A. (1998) State Transition Diagrams. In Handbook on Architectures of Information Systems. P.Bernus, K.Mertins and G.Schmidt (Eds), Berlin : Springer. pp147-166

Dick, N., Huschens, J. (1998) IAA - The IBM Insurance Application Architecture. In Handbook on Architectures of Information Systems. P.Bernus, K.Mertins and G.Schmidt (Eds), Berlin : Springer. pp619-638

Elmasri, R. and Navathe, S.B. (2000) Fundamentals of Database Systems (3^{rd} Edition) Addison Wesley, 2000

Genesereth, M.R., Fikes, R.E. (Editors) (1992) Knowledge Interchange Format, Version 3.0 Reference Manual. Computer Science Department, Stanford University, TR Logic-92-1

Halpin, T. (1998) ORM/NIAM Object Role Modeling. In Handbook on Architectures of Information Systems. P.Bernus, K.Mertins and G.Schmidt (Eds), Berlin : Springer. pp81-102

Halpin, T., Bloesch, A. (1999) Data modeling in UML and ORM: a comparison. J. of Database Management, 10(4) pp4-13

[20] seen as robust solutions to commonly occurring problems, composed of name, problem, solution and possible trade-offs

Harold, E.R. (2001) XML Bible. 2nd Edition. John Wiley & Sons
Hay, D.C. (1996) Data Model Patterns. New York : Dorset House.
Hoare, C.A.R. (1985): Communicating Sequential Processes, Prentice Hall
Hennessy, M. (1988): Algebraic Theory of Processes, MIT Press
IEEE (1998a) Standard for Function Modeling Language – Syntax and Semantics for IDEF0. IEEE 1320.1:1998
IEEE (1998b) IEEE Standard for Conceptual Modeling Language – Syntax and Semantics for IDEF1X97 (IDEFObject). IEEE 1320.2:1998
ISO (1989) Information processing systems – Open Systems Interconnection – LOTOS – A formal description technique based on the temporal ordering of observational behaviour. ISO 8807:1989 Geneva : International Standards Organisation.
ISO (1992) The EXPRESS Language Reference Manual. ISO IS10303-11, Geneva : International Standards Organisation.
ISO/IEC (2001) Information Technology – Unified Modelling Language (UML). Geneva : International Standards Organisation.
Jensen, K. (1997a,b) Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Vol 1: Basic Concepts, Vol 2: Analysis Methods. Berlin : Springer Verlag.
Krahl, D., Kittlaus, H.B. (1998) The SIZ Banking Data Model. In Handbook on Architectures of Information Systems. P.Bernus, K.Mertins and G.Schmidt (Eds), Berlin : Springer. pp667-688
Larman, C. (1998) Applying UML and Patterns, Upper Saddle River: Prentice-Hall
Mayer, R. J., Menzel, C., Painter, M., deWitte, P., Blinn, T., and Benjamin, P. (1993) IDEF3 Process Description Capture Method Report. Wright-Patterson AFB, Ohio, AL/HRGA.
Menzel, C., Mayer, R.J. (1998) The IDEF family of languages. In Handbook on Architectures of Information Systems. P.Bernus, K.Mertins and G.Schmidt (Eds), Berlin : Springer. pp209-242
Mylopoulos, J. (1998) Characterizing Information Modeling Techniques. In Handbook on Architectures of Information Systems. P.Bernus, K.Mertins and G.Schmidt (Eds), Berlin : Springer. pp17-58.
NIST (1993) Integration Definition for Information Modeling (IDEF1X). Federal Information Processing Standards Publication 184. Computer Systems Laboratory, National Institute of Standards and Technology.
Murata, T. (1989) Petri nets: Properties, analysis, and applications. Proceedings of the IEEE 77(4) pp 541-580.
Oliver, D.W., Kelliher, T.P., Keegan Jr, J.R. (1997) Engineering Complex Systems with Models and Objects (paper available on-line at http://www.incose.org/newsline.html)
Peterson, J.L. (1981) Petri Net Theory and the Modeling of Systems, Prentice-Hall, N.J.

Petri, C. A. (1962) Kommunikation mit Automaten. Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962. Also Second Edition (1966) New York: Griffiss Air Force Base, TR RADC-TR-65–377, Vol.1

Rumbaugh, J., Jacobson, I., Booch, G. (1998) The unified modeling language reference manual, Addison Wesley

Scheer, A.W. (1998) ARIS-Business Process Modelling. Berlin : Springer-Verlag.

Smith, G. (2000) The Object-Z Specification Language. Kluwer.

Spivey, J. M. (1989) The Z Notation: A Reference Manual, Prentice Hall.

Sowa, John F. (1984) Conceptual Structures: Information Processing in Mind and Machine. Reading, MA : Addison-Wesley.

van Eijk, P.H.J, Vissers, A. and Diaz, M. (Eds) (1989) The formal description technique LOTOS. Amsterdam : Elsevier.

Vernadat, F.B. (1996) Enterprise Modeling and Integration: Principles and Application. London : Chapman & Hall.

XML (2000) Extensible Markup Language (XML). Version 1.0 2^{nd} Edition. W3C (World Wide Web Consortium) Recommendation.

XML (2001) XML Schema Part 1/2: Structures / Data Types. W3C (World Wide Web Consortium) Recommendation.

12

MODELLING THE MANAGEMENT SYSTEM - ENTERPRISE MANAGEMENT AND ACTIVITIES

Cielito Olegario[1] and Peter Bernus[2]

[1] Universiy of Western Mindanao (Phillipines)
[2] Griffith University (Australia)

12.1 Introduction: The Function of the Managament and Control System of Enterprise Entities

Once enterprise management has decided on a mission[3] for the business and has determined the entities involved in satisfying the mission, it is necessary to define how these entities will be co-ordinated (managed) and controlled.
Co-ordination and control is achieved using a system of 'management and control' (also called a decisional structure, management and control system, or 'management, command and control' in defence). This system must be able to work out all necessary actual commitments / controls based on *actual* objectives / feedbacks from the managed and controlled system as well as from the environment.
In fact, it is not strictly speaking necessary that a mission be fully developed *before* the design of the corresponding management and control system is started – it is only necessary that in the end result the *mission* and the *system of management* are compatible and coherent. In practice, a change in mission will require some redesign of the management and control system.
Depending on the entity in question, there are many variations in the actual timeline of events, i.e. how small steps of mission (re)direction and management system (re)design follow one another. In this chapter it will be shown how to *describe* the management and control system, i.e. how to *design* it, and how to *verify* that this system is consistent with the mission of the entity.
Another remark that needs to be made is about the dynamics of management and control. Since co-ordination and control is achieved at any one time through a series of *actual* commitments (and associated authorities), the question is to what extent these commitments need to be determined by design, and what part can be determined by operating the management and control system?

[3] the *mission* determines the *category* of objectives that the company may wish to set for itself and thus must prepare for

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

The *designed* part of the management and control system defines a space of *potential* commitments and authorities. During the *operation* of this system some of these potentials become actual.
There are several typical types of designs for management and control systems. Some systems are *static*, meaning that the objectives the company can handle using its present management system are limited (i.e., as a result, with changes in company objectives the system of management and control needs to be changed). For example, a static system could handle quantitative changes in objectives but not qualitative changes.
As opposed to static systems, *dynamic* or *'adaptive'* systems of management and control allow qualitative changes in objectives without the need to change the way the management and control system is structured. This dynamic property is achieved though giving sufficient autonomy-, and delegating enough authority to management roles (and control system modules). This allows more freedom for them to dynamically configure the rest of the system as needed by the actual set of objectives at hand.
A structure of the management and control system is defined through:

- determining the structure of decision roles, and
- assigning them to individuals and/or automated software and hardware modules (e.g. decision support tools, control software and equipment,etc).

Given a management and control system, co-ordination of the company's activities is not achieved through management structures merely being in place, because co-ordination occurs only if roles and individuals, through transactions allowed by the design of the system, continually enter into *actual* commitments and assume *actual* authorities.
Some systems of management are designed to be *adaptive* in the sense that the static part of the decision structure allows a wide range of *possible* commitments and authorities to be negotiated between roles, thus depending on the present objectives of the company the actual commitments and authorities adapt themselves to the task (the actual objectives).
The following section will show how this 'static' decisional structure can be represented (i.e., the 'system of management', or 'system of co-ordination' put in place *by design*). Such a representation must be suitable for designing the system of management and control and should allow to analyse how the management system can operate to deliver the actual commitments and authorities at any one time.
The dynamic part of commitments and authorities is then negotiated through transactions as part of the company's operation, and these transactions need to be represented as well. This implies a need to define the 'language' of transactions and the 'performatives' of the language (the rules of negotiation and commitment in transactions). The contents (i.e., the information exchanged in the transactions) depend on the topic or 'object' being negotiated and can thus be defined using information modelling techniques (i.e. using a modelling language). For example, some transactions are about orders

and invoices, some about schedules and plans, etc. and therefore a model of these objects is needed. The information model (as discussed in Chapter 11) would then define what constitutes a schedule, plan, order, invoice etc.
In any given 'business model' (meaning a way of doing business) a number of enterprise entities may be involved, and each of these entities have a management and control system.
For example, in one-of-a-kind production (OKP) there may be a company (with its management and control system), a project (with its project management and control system), and a product (say an airplane, which also has its own management and control system). While these systems may be different in terms of their dynamic properties, and their designs may be based on different reference models, the same principles of design are valid for each of these entities and the kinds of tools applicable in their design and implementation are also similar.

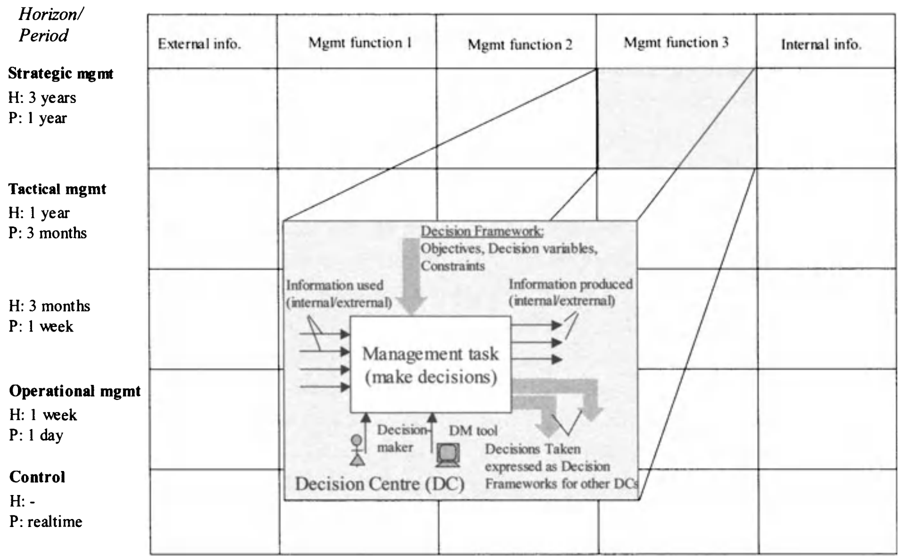


Fig. 12.1. The GRAI Grid notation for modelling the decision system [4]

12.2 Modelling the Management and Control System - What Decisions and Controls are Needed ?

The management and control system may be represented using, for example, a GRAI-grid. Figure 12.1 represents the basic terminology of GRAI-grids. The

[4] horizons and periods are intended as typical examples only

reader is referred to (Doumeingts et al. , 1998) for a detailed explanation of the 'language' of the grid[5].
The horizontal subdivision of the management grid categorises management functions (henceforward *decision* functions) according to the *horizon* and *period* of decisions. The horizon of a decision function expresses the future time interval taken into account when making decisions. The period of decision is the time interval after which such decisions are usually re-visited[6].
In addition, some events may trigger a decision function earlier then the defined period would otherwise require. Period-based decisions are typical of strategic and tactical management tasks, while on the day-to-day operational and real time levels decisions are predominantly event-based, i.e. decisions need to be made as a response to internal or external events.
The part of the management and control system that is predominantly based on predefined time intervals (periods and horizons) may be called the *management system.* The part of the system that is predominantly event based may be called the *control system.* Similarly to the management system, the control system may have several levels of decisions, but the lowest level is invariably 'real-time', meaning that the controlled process has a natural pace with which the control system must 'keep up' in order to remain in control.
The nature of decision tasks is also markedly different between management and control.
Management functions are mostly non-procedural. Therefore, in the majority of cases, no explicit procedure exists to describe how management decisions are made. However, it is possible to develop a functional model defining the *tasks* of management. Thus, management functions may be designed using some functional modelling method (such as IDEF0 models, or policies, guidelines and instructions described in natural language documents). Procedural models (such as workflows, or 'process' models) may only be developed on a very high level of granularity for such management functions. They can describe the synchronisation / co-ordination of management tasks – but one can rarely specify mandatory sequences for detailed management activities in a meaningful way. It is unhelpful to attempt the definition of a step-by-step procedure for how decisions should be made.
At the same time, functional descriptions can be developed and are very useful, because they identify the information that decision functions *must* have, and the information that they usually produce, the list of individual management tasks (functions) to be performed, the necessary capabilities of individuals who take management roles, as well as the requirements for decision support tools and databases / data warehouses.
The information links in the functional model represent reporting channels between decision functions.

[5] Chapter 3 also contains a brief presentation of the Grai-grid and a possible representation as a language.
[6] thus, decisions may be repeatedly revised

Most *discrete control functions*[7] are based on procedures (algorithms) and thus, in addition to functional models, process models can be developed that describe the behaviour of control algorithms. Such behavioural models (described using mathematical algorithms, Petri Nets, IDEF3 models, CIMOSA or FirstStep models, UML sequence diagrams, UML collaboration diagrams, state transition diagrams, state charts, etc. – as discussed in Chapter 11) describe sequences of activities, associated events, and their synchronisation although possibly with some non-determinism. Many of these models include time and/or cost, and associated statistical characteristics. Note that there do exist (adaptive) control functions that are not strictly speaking algorithmic (e.g., rule-based, neural net, fuzzy control), in which case only the interpreters may be described procedurally, with the rules serving as inputs to the interpreter. The automated part of the control system is usually implemented in computer algorithms and can be described as such. The part of the control system using humans to implement operational control procedures may be described using Workflows (van der Aalst et al. , 2002). It is possible to include in the workflow the complete procedure that is then partly implemented by humans and partly by machines.
There are two ways to analyse, design and verify the properties of a control system:

- Analysis using mathematical tools (usually employed in the process control industries)
- The production system and the control system are typically described using differential equations. In this case, the properties of the control system are either mathematically proven using explicit solutions or approximations, or the system is synthesised from modules with known desired properties[8];
- Design using simulation:
  - The algorithms of the control system are executed with assumed inputs (including assumed values at the interface with the management system and the production system) and the collected results of the simulation runs are then analysed; or
  - The control system is connected to a simulator of the production system thus providing more realistic simulation results.

Figure 12.2 shows the decomposition of the management and control system into a management system and a control system, and the languages suitable to specify (develop models of) the respective levels.

[7] Continuous control systems are not discussed in this Handbook. However, a continuous control system typically has a top layer based on discrete control (ISA , 2000) connecting it to scheduling and operational control functions of the enterprise entity.

[8] in this case only a high level model is needed to prove that the synthesised system will have the desired characteristics

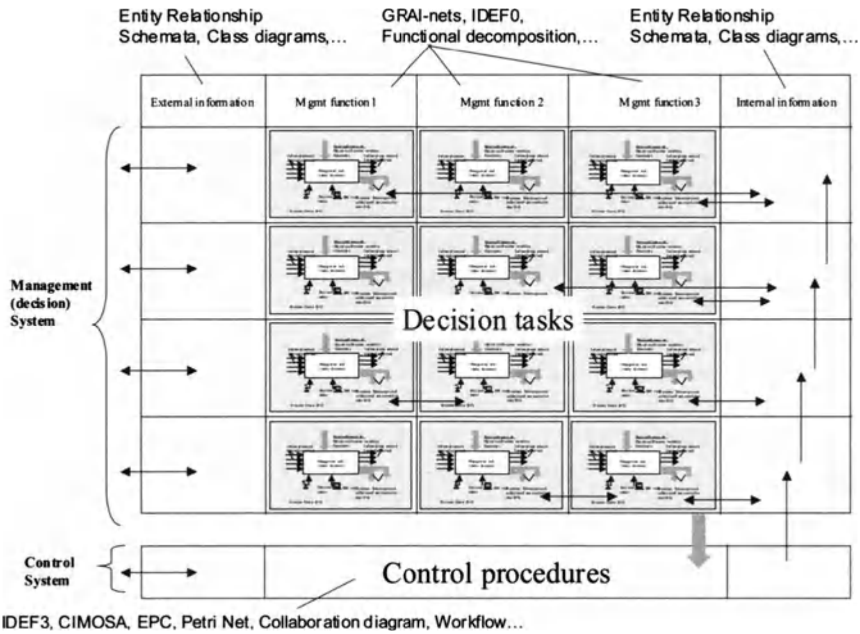


Fig. 12.2. Decomposition of the decision system into a Management system and a Control system

According to the GRAI Reference Model[9] management has three main functions: manage the inputs and outputs of the physical production system (this function is called 'product management'), manage the resources that transform the inputs into outputs (called 'resource management') and manage the consistency of decisions that are made on these two sides (co-ordination and planning).
Figure 12.3 shows the main types of management / decision tasks on a generic GRAI Grid.
Since every enterprise entity has its own decision system, there will be as many grids as there are enterprise entities. The co-ordination between the management systems of two enterprise entities can be ensured in two ways:

- One grid is operating at a higher horizon than the other (the shortest horizon of grid A is longer than the longest horizon of B). In this case A is at a naturally higher level of co-ordination then B and A has authority over B; or

[9] The generic GRAI Grid can be considered a high-level reference model for decision systems.

Horizon/ Period *	External information	Manage products (input/output)	Plan and Co-ordinate	Manage resources (human, technical, financial, information)	Internal information
Strategic mgmt H: 3 years P: 1 year		Decide what to produce in the future and from what, for whom, on what market	Decide on enterprise strategy & capabilities and core competencies, negotiate alliances	Plan resource strategy, decide on future resources	
Tactical mgmt H: 1 year P: 3 months		Decide what to produce in near future and how to reach market; Decide on suppliers	Co-ordinate / prioritise product and resource tactics	Plan resource deployment, maintenance, continuous development	
H: 3 months P: 1 week		Manage orders, procurement, and plan delivery	Production planning	Plan resource deployment, maintenance, continuous development	
Operational mgmt H: 1 week P: 1 day		Manage product delivery & incoming goods and services	Maintain production & service delivery schedule, Monitor process	Dispatch resources according to schedule; Supervisory control	
Control H: - P: real time		Real time control of resources subject to supervisory control; Provide feedback to management system			

* Note: Horizons and Periods mentioned are intended as typical examples

Fig. 12.3. Generic GRAI Grid: Typical management tasks on each level of management

- Grids of A and B are operating on the same horizon, and there are information links between the decision systems of A and B. In this case, the corresponding decision centres from A and B negotiate to agree on compatible objectives without one being at a higher level of authority then the other.

Figure 12.4 shows four entities: one corporate management entity, one factory, one product development entity, and one marketing and sales entity. The corporate management entity's mission is to provide management services to the company, therefore its service is described using a GRAI Grid; the other three entities have their own missions as their names suggest. The corporate management entity's horizons may be, for example 5 years, 2 years, and one year, with respective periods of one year, one year[10] and six months. The other three entities would have their horizons defined according to the needs of the managed process – for example one year, one quarter, one month and one week, while operational control would be exercised on two levels: daily and real time. (The corporate entity does not have direct control functions.)

[10] In this example both long term (five year) and medium term (two year) strategies are refreshed at least yearly.

[11] the mission delivery process is shown at the bottom of the figure

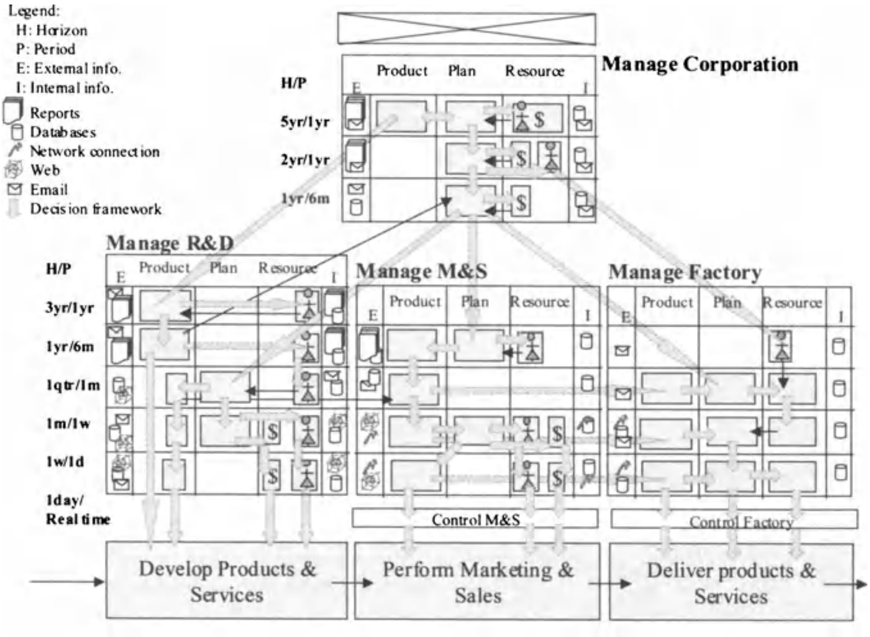


Fig. 12.4. A conventional management system: four enterprise entities with natural hierarchy and their management and control systems[11]

As can be seen from Fig. 12.4, decision frameworks are passed down the natural decision hierarchy between decision functions. The reader may find peculiar the fact that the top level horizon for the lower level entities had been defined in this example as one year (and in case of the product development / R&D entity there is a three year horizon as well) – the same or longer as the shortest horizon of the corporate management entity. This is possible (the lower level entities may have longer horizons for some decision functions as this example shows) since the requirement is only that no individual decision function should receive its decision framework from a decision function that has a shorter horizon. If, for example, the resource management task of each entity is worked out in detail, one could see that the management of financial resources on the one-year horizon is a corporate function, but at the same time, each lower level entity may make autonomous human resource management decisions on the three year and/or one year levels.
The above example shows a conventional business model, and reflects the way most corporations work today. In the rest of this chapter, other newer ways of doing business will be shown. A particular business model of interest is based on networked enterprises, where joint activities are performed in a project-like manner, in a 'project enterprise' or 'virtual enterprise' (as is often the case in one-of-a-kind industries). Virtual enterprises may also be longer lived (such

as may be the case in the joint provision of services).

Before progressing to the discussion of the management of virtual enterprises, it is necessary to go into more detail of how individual management functions may be designed, i.e. how to represent the management and control system on a finer level of granularity. Doumeingts et al. (1998) call the model of management and control as expressed using GRAI Grids a *macro* model. This model captures the interaction of decision centres (decision roles), but does not represent the finer details of the decision tasks; thus, it cannot be used directly to define the information used and the information produced by these decision centres or for the development of job descriptions for management roles.

Clearly, there is a need for a more detailed representation, to be able to design the information flows among decision functions. This includes information exchanged among decision centres, provided to the decision centres through information aggregation by a management information system (such as a data warehouse and its 'data marts' (Inmon , 2000), and information exchanged directly with entities external to the company.

One could also represent in this model the structure and content of information flows, whether they are ordinary information links, or decision frameworks[12].

The *micro-level* models of decision centres may be represented using some form of functional model, or if the decision function is procedural in nature, then using a behavioural model as well.

Doumeingts et al. (1998) propose to use GRAI-nets to represent the functional model of the management system. Another way is to use IDEF0 diagrams, and there are many other options, such as to use a tree of functions (functional decomposition) and to identify[13] for each of the functions in this tree the necessary input and output information.

High-level procedural models may also be developed to describe the synchronisation of major decision tasks (using CIMOSA process modelling constructs, IDEF3 diagrams, Petri-nets, FirstStep models, Event Driven Process Chains, etc. – refer Chapter 11), but typically the details of the lowest level activities of these process models may only be described using functional models.

For the micro-level definition of control system functions (given the procedural nature of these tasks) one could use process modelling languages as mentioned above, or (for specifying the requirements of control systems) one can use special purpose mathematical models.

To actually 'define' the content of the information 'identified' using functional or process models, one needs to go into even more detail. For each identified information flow it is possible to define a 'data schema', using a data modelling

[12] After all, a decision framework is only an information flow, from a 'superior' decision centre to a 'subordinate' decision centre – it is only that a decision framework has a defined structure, such as invariably consisting of objectives, decision variables, and constraints on how these variables may be manipulated

[13] 'Identifying' in this context only means 'naming' the information used / produced, and does not mean that a complete data model is developed for each of these.

language, such as IDEF1X, Extended Entity Relationship data model, UML Conceptual schema, Object Role Model (NIAM), CIMOSA Object model, etc. (refer Chapter 11). Whichever modelling language is chosen, the bottom line is that entity types (object classes), their relationships (associations) and attributes will be defined. While these languages have slightly different expressive power, the choice should be dictated by the availability of modelling expertise (modelling methodology) and the availability of tools to support the modelling activity. After all, on the design level, *any* of these forms may be mapped onto a language of choice, such as SQL, XML, etc.

It is important to note that modern management information system design methods advocate that Data Warehouse design should be based on a step-wise generalisation of operational database contents (databases used by the production system and the control system), rather then on functional requirements of the management system (Inmon , 2000). This chapter emphasises the need for the designer of the decision system to consider the information needs of the decision centres. This will generate requirements for the designer of the company's Data Warehouse and data marts. To arrive at an adequate- and at the same time feasible solution these two sets of requirements must be compared. One must consider what information is available in operational databases and also what information and in what form needs to be presented to decision centres. The generalisation of information content from operational databases may reveal new possibilities for decision centres (not considered by the designer of the decision system), while the requirements that result form decision system design may reveal the need to make some changes in the operational databases. In addition, the design of data marts, and especially their user interfaces must consider the decision makers' needs, in terms of usability and relevance to their decision making tasks.

12.3 Models of the Mission Delivery Process as Used in the Design of the Decision System

Chapter 11 has elaborated on the models that could be used to specify the functional requirements of mission fulfilment processes. Doumeingts et al. (1998) call this the 'model of the physical system'. Clearly, in order to design and implement mission delivery, one should understand the process that is suitable for producing the products and services of the enterprise entity in question. However, these models are also needed for the design and even the implementation of the decision system. Only the use of these models is treated here – how to develop them was addressed in Chapter 11.

Decision centres co-ordinate the activities of the enterprise. Thus, in order to know what decisions to pass down the management hierarchy, and ultimately (in case the mission delivery process is procedural) through the control system, a model of the mission delivery processes is needed.

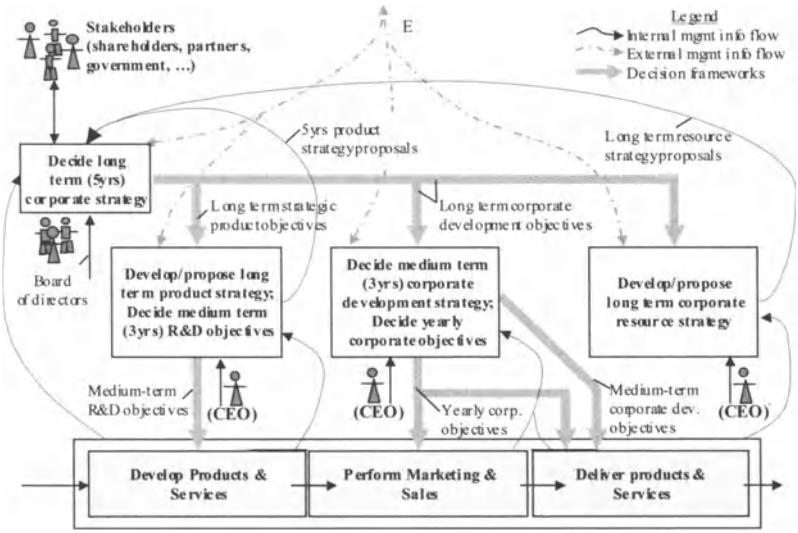


Fig. 12.5. Decision functions and their relation to the functional model of mission fulfilment[14].

The top level of this 'mission model' needs to be actually *seen* by top-level management. This view would show the entire process of the enterprise, and for a top decision centre the decision frameworks passed to lower levels would be seen as 'management and control' information interacting with this overall model of mission delivery (let's call this a *level 1* model of mission delivery). Lower-level decision centres may only see part of the above level 1 model of the mission delivery process, but the part they see should be more detailed – let's call this a *level 2* model, etc. As a result, the functional decomposition of mission delivery and the functional decomposition of the decision system are closely related. Figure 12.5 illustrates this correspondence.

The functional model of mission delivery may (although does not necessarily have to) include the decision functions as 'black boxes' (as in Fig. 12.6 b). Thus, the micro-level models of decision functions are *at the same time* the decompositions of decision centres as identified on the GRAI-Grid, and of the black-box functions of 'management and control' if they were included in the functional / process model of mission delivery.

Some modelling practitioners prefer not to include the management and control functions in the functional / process model of mission delivery (as in Fig. 12.6 a). This is done in order to keep these models readable (instead of overcrowding them with control functions). In this case, the lower level

[14] shaded boxes represent the functional model of mission fulfilment (e.g. in IDEF0); unshaded boxes represent decision functions (view from the 5yr horizon of corporate management c.f. Fig. 12.4)

controls can be introduced as 'tunnelled' arrows on the appropriate levels of decomposition. This second possibility is represented in Fig. 12.6 a.
In Fig. 12.6 a, 'tunnelled arrows' – in '( )' brackets – are used to represent management and control information, introducing the information flow on the level of decomposition, where the information becomes relevant. The tunnelled arrows are not visible on higher-level IDEF0 diagrams. In Fig. 12.6 b, lower level management and control functions are included in the functional model (shown in the figure as boxes with thick borders). Note that in base (b) the information and communication interfaces of management functions are added (grey arrows in Fig. 12.6 b). Control arrows labelled 'Long term product objectives', 'Medium term R&D objectives', 'Research project priorities' and 'Resource development priorities' correspond to decision frameworks in Fig. 12.4 and Fig. 12.5.

(a)

(b)

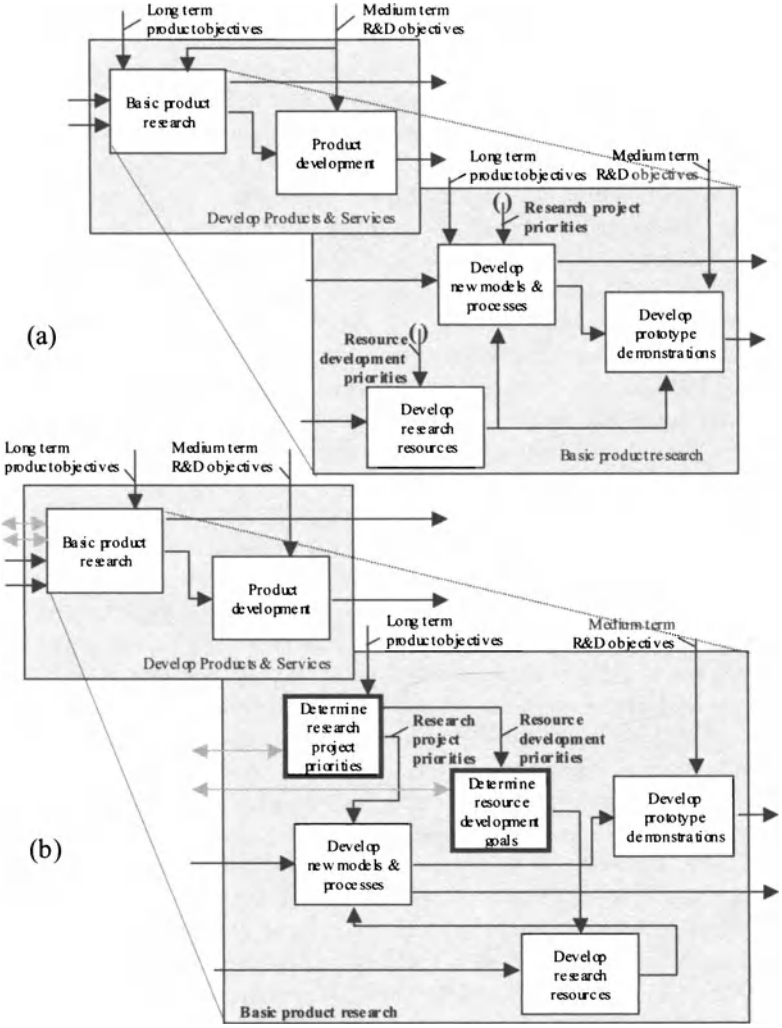


Fig. 12.6. Two ways to represent management and control in a functional model of mission fulfilment

12.4 The Nature of Decision Links - Useful Principles

At the first sight readers may be tempted to see this representation of the management and control system as a set of hierarchical relationships reminiscent of older management systems, and may wonder whether this is a relevant representation for newer modes of management where autonomy, negotiation, dynamism and individual responsibility are key factors in the success of the enterprise. However, this would be a misreading of the above sections.
First, the differences between dynamic and static management systems were already addressed, according to which the dynamism of the system is achieved through an *appropriate choice of the decision frameworks.* Thus, while it is true of any management system (static or dynamic) that strategic decisions guide the tactical ones (strategic level decision centres are superiors of tactical level decision centres, and so on), there can be tremendous differences between two management systems *depending on how objectives, decision variables and constraints are formulated.*
Consider an objective – such as the supply of raw materials to the manufacturing process in a timely manner, with decision variables only allowing parametric changes to be made. In this case, suppliers are pre-identified and not negotiable, only the volumes and delivery dates can be adjusted (decision variables being the volumes and delivery dates. Consequently the management system is rightly characterised as static. If, however, the objective is set as above, but the decision variables allow the selection of suppliers, negotiation with these, and allow planning and scheduling interactions with the selected suppliers, then the same management system will appear to be dynamic, adjusting material purchasing to the current market conditions and the requirements of the production process.
Second, the authority of decision centres (and therefore the use of the responsible manager's abilities to achieve the objectives) is also dependent on how decision roles are allocated to people. Surely, if all decision functions are performed by the same person, total authority is guaranteed, whereupon if every decision role is allocated to different people, then it is likely that no person will have enough authority, or at best, the time to reach a decision will be too long. The allocation of decision functions to individuals (or groups of individuals, like committees) is addressed in Chapters 15 and 16, where it is also discussed how this allocation determines important organisational characteristics.
Third, when designing decision functions on the micro level, there is a wide scope of design decisions regarding how to define the decision tasks of a decision centre. On the one extreme, company policies (as they exist at the time the decision system is being designed) may be interpreted as 'cast into concrete' and thus the designers of the decision task might design a decision function, or even decision procedure, that has no authority to interpret company policies: policies will be automatically complied with if the decision process is followed. While this may result in a very efficient system of management, the

result is also very brittle. Should company policies change, the decision procedures need to be redesigned. To avoid this problem, the decision procedure or function may be designed to evaluate policies (current ones, i.e. the ones current at the time the decision is to be made), and autonomously plan an appropriate decision procedure for the actual task at hand. Thus, while in rigid decision systems policies are designed into the decision procedure, dynamic decision systems allow current policies to be used as constraints (controls) of the decision function. Of course, such autonomy of a decision centre to change its internal procedures is only viable if the person allocated to the decision role has the skills and knowledge to interpret policies, and can be trusted to do this task correctly and with integrity.
Some readers may be advocates of agent based, fractal, or holonic 'organisational structures'. How do these approaches fit into the above-described way of functionally specifying management and control systems?

12.4.1 Agent-based Management and Control - the Enterprise as an Aware Agent

An agent is defined as in the Artificial Intelligence (AI) literature, based on essential, or characteristic, properties.

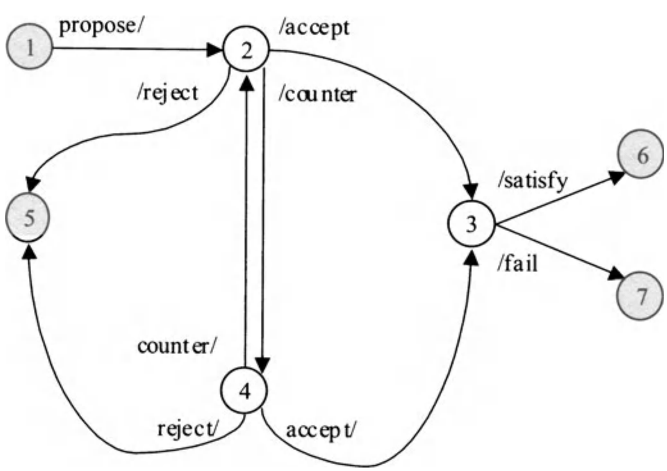
- An agent is an autonomous entity, it has certain resources at its disposal and it has capabilities, i.e. it is able to perform certain activities using its own resources;
- An agent has objectives, at any moment in time, and is committed (either explicitly or implicitly) to achieving these objectives.
- An agent is able to plan its activities in such a manner that by performing these activities according to the plan it will achieve its objectives.
- An agent has some degree of awareness, meaning that it is able to observe the progression of its activities as they are performed, so as to establish whether by performing the plan it actually is making progress to achieve the objectives, and if not, it is able to either modify its plan or modify its objectives. E.g. if one of the agent's resources suddenly becomes unavailable, the agent may have to modify its plan.
  For the purposes of this discussion, it is immaterial whether the agent actually has an explicit representation of its objectives and plans, or whether it actually performs some procedure to generate a plan or not. What matters is that for an external observer the agent appears to be doing all these things.
- An agent may be implemented as a 'substantive' individual, i.e. in some form of physical existence - either as an automated system (software and/or hardware), or as a human, or as a hybrid human-machine system. Since decision making today usually needs automated tools, decision centres would typically be implemented as a hybrid human-machine system (hybrid agent).

- An agent may communicate with other agents through 'agent negotiation protocols'. Negotiation involves proposing or accepting to do certain tasks (i.e. either agent A commits to adopt an objective as determined by another agent B, or agent A may request agent B to commit to an objective). Agents honour such negotiated agreements, and there is a protocol through which agents inform one another about the progression (or otherwise) of this agreement.

  Agent negotiation protocols are a basic form of coordinated action between agents and allow two or more agents to perform a task together that none of them would have been able to perform alone (e.g. because of the lack of available resources).

There are many variations to this definition in the AI literature, extending this list of characteristic properties, but for our purposes, the above listed properties are the essential ones.
Agenthood is a very important property in designing enterprises, because this property is desirable on every level of the enterprise. As previously discussed, agents are able to co-ordinate their activities to achieve a commonly accepted objective. Through this, two (or more) agents are able to *aggregate* into a higher-level agent, which - for the duration of the agreement - appears to be following a joint objective, planning and executing actions to achieve the common objective. The higher-level agent has all agent properties, which requires co-ordination through negotiation protocols. Some protocols designed by the AI community have become prominent, such as FIPA (FIPA , 2002). FIPA defines the 'performatives' necessary for agent co-ordination. One can think of these performatives as 'coloured envelopes' (green means 'request', 'yellow' means acceptance, etc) - with the contents of these envelopes (i.e. the topic of discussion between agents) being unique to the area of joint activity (or universe of discourse) in which the agents act. Thus, while all agents may use the same protocol (same performatives), the contents of their negotiations are very different. Thus, the protocol's 'coloured envelopes' may carry very different messages from agent to agent.
As an application of the agent model, it is desired that when two or more people form a group (committee, working party, team) the group itself should become a higher-level agent. Should this not be so, individuals in the group may not contribute to the joint objective, and this may become unnoticed, or crucial resources may remain un-utilised, joint activity may not lead to the achievement of the joint objective, etc.
Similarly, a department, a complete enterprise, or even a set of enterprise entities should be able to aggregate their lower level agents so that this aggregate entity also behaves like an agent. This is a basic requirement of any collaborative activity - between individuals, and in general between enterprise entities formed to jointly achieve some objectives.

The resulting model may be called the model of an *aware enterprise.*
An important lesson from this section is that mechanistic control models are rarely adequate for the design of the management and control system. This is because the co-ordination between decision centre A and decision centre B is not adequately described by A issuing a control to B and observing a feedback regarding the performance of the assigned task. Such rigid control has no ability to gracefully degrade or to find alternative solution pathways and except for low-level control tasks is not preferred (Bernus and Nemes , 1999). It is much more adequate to imagine the relationship between A and B as a series of negotiation transactions between A and B. A proposes to B an objective, which is followed by an agreement (possibly through iterative negotiation) in which B commits to achieving a type of objective in the future, and commit to actually achieve such an objective when it becomes actual.
A simple state diagram (Fox et al. , 2000) in Fig. 12.7 illustrates the main states of a negotiation protocol. The figure does not show all possible states and transactions (performatives that bring the negotiation from one state of the other), as for example, might be found in the FIPA protocol, but is a minimum set of states that should be the backbone of a negotiation.
It is easy to imagine that agent negotiation protocols become the norm in decision systems, i.e. every decision centre should have the same interface, in terms of the types of messages exchanged with other decision centres (with the contents of the messages of course being different from agent to agent).
Thus if a management and control system is designed based on the agent paradigm, one needs to ensure that each decision centre conforms to such a pattern, as well as each decision centre is treated as an agent, and has the requisite planning and negotiation functionality as set out above.
Note that the information technology literature uses the world 'agent' in two different senses. One sense is compatible with the definition given above. The other quite common usage of the term 'agent' is referring to software objects that are able to perform some actions at the request of some other similar agent or a human user, but the above-mentioned agent properties do not necessarily hold either for these agents or for their aggregates. E.g. 'Java aglets' (Kiniry and Zimmerman , 1997) are quite simply mobile Java language objects that are able to travel from one computing node to the other and carry their state with them. Such mobile objects may of course be suitable implementation tools for agents in the first sense, provided these objects are given full agent functionality.
Lately, for the formation of virtual enterprises, the agent paradigm has been proposed (Goranson , 1999) – where more detailed propositions are given about the internal structure of agents, specifically suitable for forming alliances in view of desirable common objectives.
Many applications of this technology exist, starting from the computer-aided design of virtual enterprises, to distributed planning and scheduling.



Legend:

States of conversation:

1. Start	2. Proposed	3. Accepted	4. Counter-proposed
	5. Rejected	6. Satisfied	7. Failed

Utterance source:

/*<performative>* : uttered by Agent A
<performative>/ : uttered by Agent B

Fig. 12.7. Main states of a negotiation protocol (example: agent A negotiates with agent B proposing that B should perform some action (Fox et al. , 2000))

12.4.2 Fractal Enterprise Models

Warnecke (1996), in the Fractal Factory proposed that the development of manufacturing systems and factories was hindered by an unjustified variety of interfaces as exist among machine tools, robots and cell controllers, factory level control systems, and corporate management systems (in reality the hierarchy may be even higher). Furthermore, different levels of flexibility exist regarding what can and cannot be controlled at each of these levels.
A radical re-thinking of these interfaces may result in a better design, so that the way machine tools, robots and cell controllers are aggregated into cells, can be repeated all the way up this hierarchy. Thus, the aggregation function (and necessary transactions between the aggregated components) might be defined in a way that lends the resulting system a *self-similar* property. After properties of geometric fractals (Mandelbrot , 1983), which result the same pattern to appear on multiple levels, the concept was named the Fractal Factory.

Apart from the standardisation and ensuing obvious benefits of uniform interfaces and transactions through which to create aggregates, there is another benefit that such designs have. If a system can be looked at any level and controlled without having to be aware of the details how the system implements its lower level functions, the complexity of management and control processes can be reduced. In addition, reconfiguration of any level becomes simpler, because if a cell (aggregate of a given level) is reconfigured, then the control system above it need not necessarily know about this reconfiguration. Thus, the resulting manufacturing system is easier to change and becomes more agile. Given that changes such as reconfiguration are frequent events in many manufacturing systems (or frequently desired but hard to achieve) the fractal model offers greater flexibility to manufacturing.
The details of fractal systems are not discussed here, but note that a further specialisation of agent negotiation protocols can be performed, so that the uniformity in co-ordination / negotiation protocols is further extended to the specific needs of forming fractal aggregates and so the aware enterprise is built on fractal principles. Therefore, the two approaches (fractal and agent-based) are complementary. It is possible to design fractal systems without the components being fully-fledged agents, and it is possible to design systems based on the agent paradigm without the fractal property, but the combination of these properties would result a superior management and control system architecture.

12.4.3 Holonic Systems

Holons as defined by Koestler (1967) are autonomous entities that can be aggregated into higher-level autonomous entities. Autonomy is defined as in the first property in Section 12.4.1. However, it is necessary to elaborate on the notion of autonomy.
For example, a factory worker together with the worker's toolbox is an autonomous entity that is able to perform certain tasks without the aid of any other entity. Or is this true? Not quite - for example if suddenly all air is removed from the factory, or food or drink are denied from the person, soon it is discovered that the entity is unable to function without support of its environment. Even if one provides all *infrastructural* support (such as food, drink, heating and cooling, fresh air, etc.) if the factory worker is asked to perform its task without interruption for a period of 100 hours, it will be found that the worker is not able to do what was asked for. Thus autonomy is a relative property, an agent is autonomous in the sense that it is able to perform its functions without any other help then the ubiquitous infrastructure, and is able to do so within a space-temporally confined space. This confinement is not limiting the agent though, because no one wants the agent to autonomously perform its tasks outside the boundaries of this confinement and without the provision of the assumed infrastructure.

Holonic manufacturing systems (van Brussel et al. , 2000) are such system where cells (and higher level units like workshops) are built on the principle that each level of aggregation should result in an autonomous agent (or holon). This holonic property is actually a useful condition if one intends to use the agent paradigm in the design of management and control systems, giving us further guidelines on how to aggregate agents. After all, not every possible aggregation will result in autonomous agents. For example, two agents might only conditionally commit to designing and manufacturing clamping devices, e.g. a tool designer and a manufacturing agent would be able to do that. However, they may not have at their disposal an assembly station that is needed for assembling the clamping device. If the aggregation is delayed to the point in time when the actual request is produced for a particular clamping device, the agent would have to go out and try to get an assembly station to commit to the requisite assembly task - which may or may not be available at the time of the request. Note that the agents do not have to go out and secure the commitment of the State Electricity Board to commit to providing electricity to power their tools, because electricity is assumed to be provided by *infrastructure.* What is or is not considered infrastructure is situation dependent - while a delivery company may assume that there will be petrol stations available as part of the infrastructure, international armed forces having to deliver food supplies to a remote area in the jungle may not assume the same thing. In summary, in using the agent paradigm for the design of decision systems it is a useful rule of negotiation between agents to seek commitment of all other agents whose services will be needed to achieve the joint objective, and form holonic aggregate agents in their co-ordinating negotiations. In fact, most research in Holonic manufacturing concentrates on multi-agent implementations. The difference between functionally aggregating the organisation and forming holonic aggregates is that in functional organisations every agent is part of one and only one aggregate, thus forming a *hierarchy.* In holonic systems, one agent may be part of more then one aggregate, and as a consequence there are many aggregate hierarchies rather then one. This type of aggregation is called a *heterarchy.*

12.5 Enterprise Building Transactions

The design of individual enterprise entities may be achieved through applying the methods and tools presented in Sections 12.1-12.4. However, how should multiple enterprises co-operate to form higher-level aggregations to achieve objectives that are not in the competence of any participant? Although the same methods and tools may be used as mentioned, there are special considerations and details, which must be kept in mind on this level of aggregation:

- While individual enterprises also consist of several enterprise entities, there is an assumption that all assets / resources of the company (financial,

human, machinery, buildings, materials, etc) are owned by the company. When multiple enterprises share parts of an objective, and to achieve it they participate in parts of a mission delivery process, all of them dispose separately over their own resources.

- Enterprise Entities within one company, being under a single corporate management, can be dictated to a certain extent and their activities fall under a set of objectives that strategic management harmonises, or at least has the mandate to harmonise. When multiple companies share parts of a process, each company has a separate set of objectives, and there is no desire or need to harmonise all objectives of all participants. What is needed is a limited agreement on a joint objective and commitment within the boundaries of the negotiated agreement. This invariably needs some concessions on behalf of the participants, partially giving up their autonomy, but certainly, this is a limited commitment, only to achieve the joint purpose. This raises a number of issues that will be discussed below (such as trust, risk, responsibility, indemnity, etc).
- To effectively negotiate agreements, companies must have commonly or jointly accepted ways of negotiating commitments, which involves transactions over and above the normal business transactions linking the operational levels of the participants (such as placing orders, accepting shipments and settling accounts). In addition to these operational level transactions, participants must conduct tactical and strategic level transactions, such as agreeing on new ways of doing business together, performing joint planning and scheduling, etc.
- While in certain industries, and in limited geographical areas, this happens as a matter of course (everybody implicitly shares a set of strategic and tactical objectives, such as companies in the Hollywood film-making 'strip', or in certain company networks) most industries do not generally enjoy the benefits of such common understanding between all participants. This is because once companies need to open up to doing business globally with companies from different cultural and business environments, many of the assumptions that allow, e.g. Hollywood companies to work together so efficiently, do not hold. Similarly, most Japanese companies, used to deal with partners from the same background and homogeneous business culture, will find that the kinds of agreements and associated transactions that were previously adequate do not scale up to global business transactions. Therefore, many assumptions that were implicit and hidden in a homogeneous business culture need to be spelled out and agreed on, before successful business transactions can be contemplated.

For the above reasons it is appropriate to continue this chapter with the discussion of global virtual enterprises.

12.5.1 Virtual Enterprise Building Transactions and the Need for Common Reference Models

The virtual manufacturing enterprise (VE) is a temporary alliance brought about by the co-operation and collaboration of multiple enterprises with complementary core competencies. A Virtual Enterprise appears as a single entity to the customer whose demands it seeks to fulfil (Georgakopoulos et al. , 1999; Bernus and Nemes , 1999; Klenke , 1996; Hardwick and Bolton , 1997). Classic examples of virtual enterprises are those ad-hoc arrangements in the movie industry (Goranson , 1999; Aerts et al. , 2000) as mentioned in the introduction to Section 12.5. Because of the way information technology enables enterprises to operate across time and space and the fast rate at which global markets are growing, customer demands include not only quality and cost-effective products, but also speed in the delivery of the required goods or services. Addressing the demand through resource and risk sharing with enterprises in the same supply chain may become more viable. VE arrangements allow the possibility of venturing into new markets for a company that lacks R&D capabilities but has a core competency which makes it an attractive potential partner for a company that conducts appropriate R&D.

Aerts et al. (2000) distinguish the VE from an integrated supply chain by stating that an integrated supply chain depends on a more long-term inter-organisational structure catering to a relatively stable market. As opposed to this, a VE is opportunity-driven and usually dissolves once the demand has been fulfilled. Hence, it "defines itself by its specific linking" (Wiedahl et al. , 2000). However, it is very possible that a VE arises from an integrated supply chain with the assimilation of temporary partners/suppliers into an alliance to address an opportunity (Vesterager et al. , 2000; Zhou , 2000). In this case, the principal actors (*(partners)* would belong to a *network* and the *virtual enterprise(s) would be created by the* network.

A VE is established similarly to a real business enterprise where the partners' core competencies and relevant resources (human, financial, etc.) are shared (Klenke , 1996). The VE partners come together with the sole focus on cost-effectiveness and product uniqueness without regard for organisational size or geographic location (Hardwick and Bolton , 1997), and Nemes (2000) states that to enhance the operational agility and global networking capabilities of manufacturing firms these firms "need tools and methodologies for enterprise management, co-operative planning, concurrent engineering, adaptable production automation, decision support and interaction protocols for operating virtual organizations". He identified critical areas of interest for these manufacturing firms, citing among them the design of the extended enterprise, network information management and computer-supported collaborative work.

Bernus and Nemes (1999) argue that the virtual enterprise that operates in volatile environments must address the need for *agility* in setting up and operating, the need for *flexibility* in adjusting to the independent cultures of the members who will comprise it, and the fundamental question of how

seamlessness can be achieved. Given that a virtual enterprise brings together independent companies, each with its own culture and each with its own way of operating, it is important to have a shared understanding of what are the goals of the virtual enterprise and how planning, joint operation and eventual dissolution can, or will be, done.
Bernus (2001) states, "in order for a set of companies to be able to jointly design and create a virtual enterprise, it is necessary for them to have some level of previous agreement on how to do business together". For dynamic organizations specifically, it is imperative that there be commonly available proven reference models for such a scenario (Bernus and Nemes , 1999). A model (a reference model in this context) is defined as "... any construct, on paper, in the computer, or any other medium, which shares some important properties with a real or contemplated system" (Bernus , 2001).
Other researchers have also stated this call for reference models. For instance, Vliestra (1996) states: "enterprise business modelling is a prerequisite for enterprise integration". Uppington (1998) supports this view by affirming that, "it is important to have current business models in place which depict these processes. These models then promote the accessibility of knowledge within the enterprise". There is a great need therefore in the provision of a reference model that is a high level abstraction of management and control describing co-ordination links between the actors in a Virtual Enterprise – including the partners, the network and the virtual enterprises created.
It is important therefore to identify the interactions occurring between the partners, the network and the virtual enterprise formed and in so doing, to identify the decision frameworks which define the boundaries of the interaction. This reference model of management and control activities can serve as a blueprint on how decision activities concerning planning, scheduling, controlling and operating a virtual enterprise can be synchronised and co-ordinated. The role of the reference model will therefore be to provide the basis for shared understanding between the companies who shall come together to form the virtual enterprise and will therefore allow the virtual enterprise to address the customer demand in an agile fashion.
Such a reference model must describe, to a sufficient level of detail, the processes and associated information- and resource requirements for creating and operating VEs, so that the model can be used for the analysis and design of the effectiveness and efficiency of doing business together in the given way[15]. The scope of the management and control reference model should extend to product-related, resource-related, and co-ordination and planning activities. It encompasses decisions that will have to be made - not just on the strategic and tactical levels, but also those that have to be acted on a daily basis to en-

[15] Note that this model does not include the actual production and service delivery functions: a model of the necessary production and service delivery processes to be jointly delivered is a *pre-requisite* to being able to design this reference model – refer Chapter 11

sure seamlessness. The model must also be generic enough for potential virtual enterprise partners to be able to identify themselves in the model provided. In the sequel, it will be assumed (to be able to discuss an example), that a typical one-of-a-kind product's bidding, engineering, procurement, construction and after sales service is the scope of the network and its VEs.

In the process of developing a reference model for the management and control of activities occurring in the virtual enterprise scenario, one must clearly describe the generic interactions between the actors. Further, there must be an understanding of how synchronisation and co-ordination of these interactions is accomplished through the identification of the objectives, decision variables and constraints of the decision activities.

The emphasis is therefore in seeking *to identify* what activities require joint planning, joint decisions and joint actions. In so doing the actors involved in these activities are identified and the required management capabilities are determined. Since the nature of partner alliances may be diverse, such a reference model must express inter-enterprise activities that are generically shared by all types of virtual enterprises (at least, in our example, in one-of-a-kind production). A GERAM-based VE Framework (Extended Enterprise Scenario) of the Globemen[16] Consortium may be utilized to analyse the life-cycle activities/processes of the involved entities, and to determine the generic interactions between these. The extended enterprise scenario applies the principle of 'recursiveness' where the virtual enterprise is a product of the network and the network is a product of the partners. GRAI Grids will be used to express the intended reference model and an analysis of each of the decision centres will be provided.

12.5.2 Crucial Considerations for VE Creation

The Virtual Enterprise Framework initially introduced by the Globeman 21 consortium, and further developed by the Globemen Consortium allows for the representation of all life-cycle activities of all involved entities for a Virtual Enterprise Scenario. The framework also allows the representation of the interdependence of these activities. The lifecycle representations (in GERA, as shown in Chapter 2) are subdivided into two purpose views, representing

- management and control activities, and
- production activities and/or services to the customer.

In the VE Framework, three separate entities will be in the focus of discussion (more specifically the management and control systems of these entities).

Baltrusch (2000) has also developed a functional model (expressed in IDEF0) representing these interactions: including the formation of networks by partners, the formation of VEs by the networks, and the management of the

[16] Globemen: Global Engineering and Manufacturing in Enterprise Networks (Globemen , 1999)

VEs to deliver products of services. Individual functions of this process are of course identical to the functions of the decision centres to be in place in the respective three entities.
Referring back to the VE framework, one can see that there is interdependence between the lifecycles of the three entities, and it is logical to assume therefore that it is possible that the decision framework for a specific decision centre of a given entity will be formulated by the decision centre of another entity. Cardoso et al. (1999) identified some critical areas for co-operation, which are: technical data exchange, concurrent engineering, joint planning, and consortium formation/partner selection. It is necessary to investigate the specific issues raised in the literature, issues that influence the success or failure of co-operative relationships in a virtual enterprise. Therefore, before going into the detail of the decision models (the decision centres and their interdependence), in Sections 12.5.2.1 - 12.5.2.4 some crucial functions are singled out that need special attention in VE creation.

12.5.2.1 Partner Selection and Consortium Formation

For the virtual enterprise to be an attractive proposition for doing business together, partners need to have complementary core competencies. Lorange and Roos maintain that beyond core competency, there must also be a match between the strategic intents of the different partners (Lorange and Roos , 1992). Bernus and Nemes concur by stating that a harmony of objectives must be achieved to dynamically create and sustain integrated virtual enterprises (Bernus and Nemes , 1999). Companies may have a plethora of reasons for joining an inter-enterprise alliance. One company may join an alliance to be able to gain a niche in a previously inaccessible market due perhaps to the lack of resources (manpower, R & D, capital). Another may join an alliance simply to maintain its position in a volatile environment.
Lorange and Roos (1992) argue that for a strategic alliance to be successful, the strategic intents of the participating partners must be reconcilable. They suggest some guideline questions (Ibid: 30) that must be considered carefully if a 'win-win' situation is desired:

- What are the broad, readily apparent benefits from this alliance for each partner?
- How important is the strategic alliance within each partner's corporate portfolio?
- Are the partners leaders or followers within the particular business segment?
- Do they combine to create strength in an offensive manner, or is this a case of the sick joining the sick?
- Are the partners sufficiently similar culturally?

Of course, these are not the only questions that must be asked but certainly, the goal is for each partner to be able to be transparent in its intentions in

joining the alliance and in its representation of its core competence so that the other prospective partners may be able to assess their own strategic match. While not all of these questions need to be answered in favour of creating an alliance, if one of these factors is a contra-indicator, it much be carefully addressed and the associated problems need to be solved.

When a company begins to assess its strategic match with other companies, it does not look only outward, but examines the impact of the potential partnership on both external and internal stakeholders. It is crucial especially, to weigh up the support of key internal stakeholders who can make or break the alliance early on (Lorange and Roos , 1992). Macmillan and Jones, according to Lorange and Roos (1992) suggest that management must aim to (1) better understand the internal stakeholders' behaviours and methods, and (2) handle internal coalitions successfully – if the inter-enterprise alliance is to be given a fair chance to succeed. Similarly, due consideration must be given to political intentions of the external stakeholders (owners, board of directors, etc.). It is therefore essential that the overall reason for joining the alliance is understood by key individuals and groups who shall be affected and proper support mechanisms must be provided for the transition from the as-is to the to-be situation.

For instance, the 1987 Booz-Allen and Hamilton Best Practice Survey of Alliances (as cited by Freidheim (1998)) suggest that there must be a plan for relationship and communication building on all levels. They consider the establishment of frequent formal and informal meetings as a future best practice, as well as the provision of other avenues of communications (email, lists) on key levels. Jointly agreed upon policies and procedures that are documented and disseminated as needed will also help foster the requisite common cultural elements, which ensure that the VE partners have the same priorities and values for the project concerned (Bernus , 2001).

12.5.2.2 Trust

An investigation of the literature on virtual organisations contains common social themes: agility and flexibility, collaboration and co-operation, trust and commitment (e.g. (Bernus and Nemes , 1999; Goranson , 1999; Harris et al. , 1998). One profound meaning of trust in the context of virtual enterprises comes from McAllister (cited by McGrath and More (1999, p592)), who states that "Trust is the degree to which parties are willing to act on the words, actions and decisions of others".

Trust for virtual enterprises is two-fold. The VE partner relies on its trust in the network and in its virtual enterprises to treat it fairly. Further, the virtual enterprise will have to rely on trust that the partner will fulfil its obligations in a timely and agile way.

Intangible relationships such as trust relationships play a significant role in ensuring that the alliance is successful. Many researchers for example, Nandhakumar (1999) and Harris et al. (1998) argue that trust relationships

are required to address 'virtuality'. Studies conducted by McGrath and More (1999) have shown that lack of trust contributes significantly to a gradual degradation of performance of co-operative alliances.

Basic trust is secured in a community sharing traditions (Nandhakumar , 1999). Goranson (1999) calls this type of trust 'inductive trust'. Inductive trust is nurtured within everyday routines and predictability (Nandhakumar , 1999; Goranson , 1999). But what of the virtual enterprise which brings together independent members with different traditions? The importance of achieving trust becomes even more pronounced because enterprises engaged in virtual enterprises might have significant investments at stake.

Goranson (1999) argues that for the virtual enterprise, partners will have to rely on another form of trust, called 'deductive trust'. He sets out deductive criteria, which define the 'trustworthiness' of a partner. He argues that a virtual enterprise partner can be trusted if the partner is accountable, timely, accessible, accurate and complete. The partner must be able to "know fully and represent honestly the capabilities of the supplier" to the virtual enterprise's preferred level of detail. The partner is timely and accessible in the sense that the virtual enterprise can get what it wants, when it needs it" (Ibid). While it may seem unrealistic to expect the virtual enterprise to get what it wants and when it needs it every time, partners must still strive to attain a 'reasonable' response time, given the time and distance constraints.

In addressing the question of different traditions and organisational cultures coming together, virtual enterprises rely on abstract structures and processes to foster and nurture trust relationships. Among the abstract processes, the literature frequently mentions the use of legal structures in addressing the issue of legitimacy and guarantees of expectations ((Nandhakumar , 1999; Goranson , 1999; Spinosa et al. , 1998)). In the configuration phase of the Virtual Enterprise, the partners are pre-qualified or pre-selected, hence the setting for collaboration is laid out and the legal contracts are negotiated and drawn.

The members of the virtual team members therefore act within the ambit of legal contracts, which for instance may clearly define the roles and rights as well as indicate in black and white the level of commitment of the partner-member. Some trust literature authors (e.g. (Handy , 1990; Shaw , 1997) argue that trust requires boundaries where each work unit energetically and effectively works to solve its own problems using the freedom allowed by these boundaries. The 'legitimacy' of the operation addresses this need for boundaries.

It should be noted, however, that studies on strategic alliances and virtual team working conducted by Harris et al. (1998) and Nandhakumar (1999) have shown that relying solely on abstract processes does not address the very human need for psychological rewards. According to Klenke in order for network organisations to work, bonding must be achieved that allows the goals of the smaller members to identify with the network as a whole (Klenke , 1996), a view supported by earlier work of Handy (1990). There is an emphasis that

bonding begets trust and to accomplish this, trust relationships should first be established through face to face contacts and supported later by relevant information and communication structures that nurture the basic trust. It can be argued therefore that the virtual enterprise must invest in establishing actual contact among the members who will comprise the virtual enterprise, and this can also be facilitated by technology (e.g. groupware). The virtual enterprise members will therefore have something on which to build their trust relationships.

Cooprider and Victor (1993) argue further that repeatedly facilitating activities that encourage communication and social interaction as well as goal attainment provides the environment where understanding of the other partner's realities is enhanced. Activities such as joint planning sessions, joint training sessions on inter-enterprise tasks (engineering, management, some areas of sales and marketing) are some suggestions. These activities provide the atmosphere where issues can be brought out and resolved, as well as provide a common direction despite the diversity in company interests.

Enterprise relationships based on trust and commitment lead to effective sharing of knowledge through collaboration and co-operation. A well-established collaborative relationship makes flexibility and agility possible. If an enterprise can nurture and support these essential relationships, then the foundation for seamlessness, or 'operating as one', will have been laid. If the partners (and the workforce they bring with them), share a common understanding of what is to be accomplished and how it will be accomplished, trust will be secured in the virtual enterprise.

12.5.2.3 Level of Maturity

In the pre-qualification phase, where network management determines who among the network partners shall comprise the virtual enterprise for the demand at hand, the level of technological maturity of the potential partners is a significant selection criterion. The actual acting groups of the selected partners must therefore be *representative* of the respective partners' technological maturity that was presented to the network during the selection process (Pedersen and van den Berg , 2000).

Technological maturity covers both *process* metrics (what processes do prospective partners follow and to what extent?) and *product* metrics (how do products and services measure against the state of the art). Process metrics are either available through service agencies that do process assessment, or the assessment can be done by principal partners against international standards (or accepted industry standards). In the software engineering industry, for example, one might use the Capability Maturity Models (Carnegie-Mellon SEI , 2003) of the Software Engineering Institute, or the SPICE (ISO/IEC , 1998b) model (Software Process Improvement and Capability Enhancement), or the standard ISO 12207 (ISO/IEC , 1998a). Engineering companies might use

ISO 15288 (System Life Cycle processes) (ISO/IEC , 2002), which is a standard describing necessary processes and their outcomes in systems engineering. Consulting companies may use ISO 15704 (Requirements for Generalised Enterprise Reference Architectures) (ISO/TC184/SC5/WG1 , 2000) to assess the completeness and scope of their methodologies.
Similarly, maturity metrics such as the European Foundation for Quality Management (EFQM) model allows the assessment of the maturity of the management and control system of a prospective partner (EFQM , 2000). The result of such assessment may shed light on missing or inadequate management functions that may be vital for the partner's success in the network and the VE.
Product and service maturity is easier to assess, because products and services provided by a potential partner are visible – unlike processes. However, in partnership formation, one needs to also look at the prospective products of partners, i.e. what product developments are in the pipeline, and this needs consultation and some level of transparency when partners negotiate.
These maturity properties affect the specification and design of the network and of its VEs. Specifically, they directly impact interoperability (C4ISR , 1998), both technology- and management-wise. The impact of maturity across the network has a direct influence on technology choices, such as the ability to choose a flexible Information and Communication Technology (ICT) architecture, and information management and supporting tools for the whole life cycle of the virtual enterprise (according to Globeman21 (1999) and Camarinha-Matos and Afsarmanesh (1998)).
The level of maturity will also influence the agility of the network and its VEs: different management styles need to be reconciled as well as the organisational structures of the entities involved might need some adjustment ((Lorange and Roos , 1992; Freidheim , 1998; Globeman21 , 1999)). In the authors' experience, some companies had to change their organisational structure to be able to make internal responsibility structures visible and controllable. This can be interpreted as a measure taken to improve the maturity of their own management and control system.

12.5.2.4 Autonomy: The Distribution of Decision Rights

While it is true that the VE partners are engaged in a collaborative alliance, a successful virtual enterprise allows its members to maintain an adequate extent of autonomy (Bernus and Nemes , 1999). Unnecessarily detailed control can hinder a partner's ability to draw upon its own resources that would have otherwise optimised resource usage (Ibid). This autonomy should be defined at the network level and disseminated to the work force level as needed. Bernus and Uppington state that it is crucial "to define the domain where autonomous decisions should take place, and clearly identify the points where harmonisation of individual interests must occur" (Bernus and Uppington , 1998). In a sense, the organization is giving up some of its autonomy to be

able to achieve its strategic intent. A clear definition of the boundaries of independence allows controls to be implemented - not just at the network level but disseminated throughout the partner organizations as well. Provisions must also be made for conflict resolution (Ibid).
The adoption and utilisation of *jointly agreed upon information exchange processes and network policies and procedures* provides the context in which autonomy evolves. A partner's own operational resources and capabilities form a natural limit as well (Bernus and Nemes , 1999). Policies on legal / contractual issues, joint planning at the corporate level for the network, and jointly agreed upon processes, 'common ways of doing things' as well as agreed performance indicators provide the constraints (Emery , 1971).
Information systems strategic literature (such as (Earl , 1989; Frenzel , 1992; Angell and Smithson , 1991)) suggest the use of a steering committee composed of representatives from different partners (and their involved departments) affected by the implementation of an information system. It has been emphasized that this type of committee must seek a balance between the 'experts' and the 'ludites'. Consider, for example, the network management preparatory team tasked with the development of the integrating information and communication infrastructure and with the definition of other network requirements for the virtual enterprise. To be successful, this team must strive for a balanced representation from the different network partners who will be potential members of the virtual enterprise(s) to be formed by the network.

12.6 A Reference Model for Creating and Sustaining Virtual Enterprises

12.6.1 Introduction to the Reference Model

The reference model is presented as three interlinked GRAI Grids that represent the decision systems of partners, the network and of VEs (refer Fig. 12.8). In order to keep the model small, only those management tasks are represented for the partners, which are of relevance to the enterprise network and its VEs. In the application of the reference model, these partner management roles need to be created by each partner, or – which is more likely – existing partner management roles need to be extended with the additional tasks. In the following the model is discussed in detail, and descriptions are given for each decision centre in the model. Given that each network is different, there is no representation made here that the particular reference model below is fit for the purposes of any particular network; the model is an example of the *kind* of model that partners need to develop defining tasks for partner management, for the network and for its project enterprises.

12.6.2 Overall Model of Co-ordination Between Enterprise Entities: Partners, Network, Virtual Project Enterprise

Chapter 4 has discussed in detail the considerations to be made by strategic managers of each partner in order to develop a successful company strategy (the strategic manager's role is represented as 'decision centre PC1' in Fig. 12.8). It is usually one company that plays the role of the champion, and seeks the co-operation of other companies in order to jointly implement a strategy that is beneficial to all involved partners. Thus, negotiations start with convincing prospective partners of the promise that network formation holds. The outcome of this negotiation is the identification of the network as a strategic business opportunity, and the development of the concept (mission, policies) or the network that will guide the network's design.

The type of network described here assumes that it is formed by a relatively small number of strategic partners, therefore the partners who will form the network must jointly formulate the objectives and constraints for the design of the network's *master plan.* This, as a decision framework (Fig. 12.8, arrow 1.1), acts as a 'user requirements specification' for the network and includes information referring to external factors (risks & opportunity identification, market situation) and internal factors (internal capability assessment, result of feasibility studies). This specification must also refer to any assumptions to which the individual partners have agreed, e.g. jointly agreed-upon policies, commitment of the partners to realign their internal strategies – including mission and objectives – in harmony with the other participating partners. Thus, partners must have identified the creation of the network as a joint strategic objective, and the result of this agreement is the input for the development of the network's master plan.

The responsible authority for the development of the master plan may be, for example, a *network planning team* that gets its mandate from the partners and plans the management and the operation of the network accordingly. Resource allocation to the planning team must also be agreed on by the partners, and the governance and supervision of the team's work needs to be determined. In case of a complex network, the planning team may operate as a project, with its own project management structure, similarly as described in Section 12.6.5 (i.e. as a Project Enterprise). It is a possible strategy to envisage that the network planning team should later become the management team for the network. Apart from the buy-in and understanding of this team, such arrangement is one of the possible guarantees for common understanding in the network.

At the same time, the realignment of mission and objectives will also provide the constraints and criteria (Fig. 12.8 , 1.3) by which capability adjustment needs to be planned by each individual partner. Feedback on the strategic level from capability assessment and alignment to mission realignment ensures that partners develop a feasible network strategy, taking into consideration the associated risks.

Partner negotiations are likely to result in decisions that influence the individual company's product strategy. In fact, it is possible that it is the strategic considerations about the company's products that play the initiating role in starting partnership negotiations in the first place. Similarly, each partner may separately assess its product strategy and provide feedback to mission realignment to make sure that the negotiations with other partners leverage, rather then damage the company's product strategy.
It is therefore assured that any agreed policies and assumptions that require changes on the partner level are realistic. Thus, individual partners may individually and independently develop plans for internal change projects to be able to meet the network's requirements in the future.
When developing the concept (mission, policies) of the network and defining the requirements that it must satisfy, the partners' strategic management must agree on "what the network should be able to perform in its operation phase" (Globeman21 , 1999). Thus, the outcome of this negotiation must include:

- The identification of the type of products or services that the network is envisaged to support, including the product- and project- life-cycle processes to be covered by the network, and
- Various associated requirements, encompassing all areas of network operation such as performance targets, contractual-, organisational-, ICT- and human relationship requirements; as well as knowledge, quality and environmental management needs (Ibid). Issues of the division of responsibility, costs and profits as well as intellectual property rights (IPR) must also be agreed on (Ibid), (Freidheim , 1998; Goranson , 1999).

Division of responsibility requires jointly agreed upon policies on the network's business processes and assignment of authorities to those performing these processes, to prevent potential role conflicts. The setting of performance standards contributes to consistency and integrity throughout the network. Thus, partners have to address the standardisation issues; more specifically, the standards the network should conform to. The issue of the ability to choose standards as opposed to mandatory standards must be addressed as well (Globeman21 , 1999). Requirements on costs and profits also have to be worked out. Freidheim states: "the partners need to agree which indirect and overhead charges will be allocated to the alliance" (Freidheim , 1998, p133). Contractual requirements, for instance, encompass questions on profit and cost division, IPR, and future partner selection (Globeman21 , 1999). Finally, it must be determined how partners will share responsibilities to the customer, including legal and customer support obligations. The latter will be a determining factor in the 'face' or 'image' of the alliance to the outside world. Undoubtedly, one of the most important requirements definition that needs to be done is in the statement of requirements for the network's integrating information and communication infrastructure. Forging co-operation links between independent companies is not something new. However, the extensive

utilization of ICT to support the exchange of information is one of the characteristics of the virtual enterprise concept. Hence, a "first rate communications systems" (Freidheim , 1998) or more significantly, an appropriate information and communication integrating infrastructure is of the utmost importance to contribute to the qualitative and quantitative functional adequacy of the network, through which a much quicker and accurate information exchange is possible compared to traditional means.

While there is a considerable advantage for shared risk management in organisations operating in volatile environments, complex relationships also arise because adjustments will have to be done internally, as well as externally. Adjustments will have to be made as a result of changes brought about by the interfacing of the VE partners' management and control functions, processes, skills and material flow. Thus, in assessing the requirements for the network, the partners themselves look at what requirements they have to address internally. To do this, the partners must have a working knowledge of the models and protocols that the network may adopt.

Given that operating in a network requires that many functions traditionally under the direct control of the company should be autonomous, new performance measurement tools must be envisaged, to ensure that "lost management functions are substituted to retain control of the organisation"[17] (Bernus , 2001).

As a result of partner negotiations network planning may be carried out in relative autonomy by the allocated network planning team (Fig. 12.8 , NC1). Based on its mandate (Fig. 12.8, 1.1), the team[18] can begin to formulate the master plan of the network (refer details in Section 12.6.4.1).

The network planning team utilises three decision frameworks (2.1, 2.2, 2.3) to manage and control the operation of the network. The network master plan provides the objectives and constraints in the development of an overall project plan for the intended project enterprise(s) (VEs) which may be a project enterprise for bidding enterprise, or a project enterprise for construction or another engineering or building enterprise entity (2.1). Whatever tactical decisions are made regarding various aspects of the project enterprise (e.g. methodology, overall business processes, knowledge management), they will have to be within the mandate of the partners as outlined in the master plan. All strategic product decisions on behalf of the network and the subsequent scope planning operate within the boundaries (2.2) set by the network's management (NC1). Scope planning and strategic product decisions use external information such as industry and customer data, and give feedback to the decision centre responsible for the development and maintenance of the network master plan (for adjustment if needed). Network management also determines the decision frame (2.3) for strategic decisions regarding over-

[17] I.e. the network and its VEs.

[18] or network management, if the task is to modify or update the master plan of an existing network.

all network resource planning (NMR1) – e.g. human and financial resources, project tools, etc. In summary, network management ensures that there is proper co-ordination between strategic product decisions and strategic network resource planning, by harmonising the overall objectives of the network with the network's product- and resource management objectives.

Project plan development (NC2) gets its input from client request handling (NMP2), and creates a project plan for the project enterprise that will deliver the solution to the client (e.g. for bidding, design, procurement or construction). Project plan development will act within the decision framework (2.1) provided to it by network management.

Depending on the original intentions of network partners, project plans may be developed on a case by case basis, but it is more advantageous if at the time the network master plan is developed, the network planning team also develops a reference model (or reference models) for its intended types of project enterprises. In this way, the network will be able to create adequate project plans in a short time. These reference models would include a reference model for project enterprise management (or a small set thereof) and several reference models or blueprints for the operational part of typical project enterprises (such as blueprints of how to develop a bid, or how to do collaborative and concurrent engineering, procurement and construction, or after sales service). The particular project's plan is therefore developed as a specialisation, or tailoring, of these blueprints.

Given that project management and control is very similar across a wide range of virtual project enterprises, it may be sufficient to have one common blueprint for management and control. However, if the network wants to be prepared for projects of very different size and projects of substantially different risk levels, it may be necessary to maintain a small number of project management reference models instead of only one. This is the case both for small as opposed to very large projects, and for highly innovative as opposed to secure 'run of the mill' projects.

Practice shows that there exists a set of management and control tasks that is dependent on the type of engineering activity that is being managed and is hard to express in an entirely generic form (Kalpic and Bernus , 2002). This part is called 'technical management', and usually is similar for large and small projects, as well as for projects with different levels of risk. Therefore, separate blueprints may be maintained by the network for technical management activities, typical of products and services in the network's scope. Such a technical management model can be plugged into the generic project management model(s) according to the virtual project enterprise currently being designed.

Project plan development will also guide the decision processes responsible for procurement planning for the project enterprise (2.5) as well as for combined network resource capability management (2.6). Procurement planning and network resource management *aggregate* the needs of planned projects and

within the boundaries of the their respective decision frameworks, optimises the performance of the network.

Further, the project plan developers provide the decision framework that will indicate performances (e.g. production costs, level of quality, delivery times) that must be maintained to guide the scope adjustment of the network partners who shall be participating in the project enterprise (1.9). In other words, scope planning and adjustment at the partner level is guided by the requirements (e.g. quality management) determined at the tactical level of the network. This is interesting because partners give up some of their tactical management autonomy but this limitation of autonomy is within the boundaries of the strategic agreements arrived at when determining the network's overall decision framework. At this stage, only planning level commitments are needed from the partners. Scheduling of resources and the provision of products and services will be done later, at the request of the project enterprise. Thus, requests from the network to the partners are seeking commitment to perform certain activities and to allocate resources in the *future*. Actual requests will be delivered by the project itself. Alternatively (depending on how tight the network is tied to the partners who created it) the network may negotiate resources and services with outside service providers. The amount of autonomy on this level depends on the partners' strategic intent: if the market demands very fast response, partners would have to give up some of their autonomy which is a price paid for more dynamic market response, and therefore the ultimate shared benefit of partners.

The adjustments in budget as well as the project evaluation will rely on the jointly agreed upon processes and policies in the project plan development (2.4) in order to adequately respond to reports or requests coming from the project enterprise. This decision centre (as may be implemented by a project supervisory board) in turn, guides budget allocation, or reallocation as well as other resource management and deployment in terms of IT or project tools as well as the assignment of relevant authorities (2.7).

The project enterprise will start operating at the tactical level, because its strategic direction has been mandated by the network enterprise that created it (2.8). Product-related decision processes at the project's tactical level base their initial planning activities on external information, such as the details of client requests. At this point, there is a need for co-ordination with the project's resource management. Resource management will present to product management a list of detailed alternatives for resources that the involved production and service activities will need, but it is project management that will decide what optimisation criteria to use for the selection of the best alternative, and finally give the go-ahead signal to proceed with the selected project delivery activities (3.3) and resource alternatives (3.2) in the final definition of the project's processes and resources.

It should be noted that detailed project budgeting would observe the constraints as allocated by the network to the project (2.9.1). This holds true for the workload distribution as well, which gets the authority from the assigned

network representatives (2.9.2)[19]. The control of incoming and outgoing goods and services (VMP3) will provide feedback to the decision centre responsible for the monitoring of progress. In turn, the decision framework (3.4) for the day-to-day control of the project's external interactions with the client and suppliers (VMP3), as well as the decision framework (3.5) for the operational level control of project resources (VMR3) will be provided by the decision centre responsible for scheduling and for the monitoring of progress (VC3). In Sections 12.6.3 - 6.5 each enterprise entity's management and control system is presented in more detail.

12.6.3 The Partners

Consider Fig. 12.8, representing the *macro model* of the typical decision activities for partners joining the network. In reality, there could be multiple grids representing the individual partners with slightly different management and control activities. However, for the purposes of the partial enterprise model, the partners will be represented by one typical grid, representative of decision activities that any typical company would undertake. As previously noted, only those decision tasks are represented on the partner level that are directly relevant to Network and VE creation.

12.6.3.1 Realignment of Mission, Vision and Objectives / Negotiation of Network Partnerships (PC1) as Undertaken by Individual Partners

This decision centre is at the highest stratum of partner management. Strategic management needs to continuously identify change opportunities or imperatives. Once the vision of an enterprise network is proposed as a promising way of doing business in the future (see Chapter 8), management should investigate the consequences of such a change. Uppington (1998, p336) suggests the use of an 'enterprise scan' to assess the need for change:

- "The organisation's current status quo (how it is currently operating in terms of market forces, present competitive position, and what is its projected future position);
- Whether there is a need for the organisational structure to undergo a change process;
- Whether there is a viable opportunity within the organisation to successfully implement a change process;" (refer also Chapter 9)
- "What are the objectives and resources of the change process;
- How to best achieve successful benefits from this change process; and

[19] Note that project autonomy is preserved, because the negotiation (between NMR3 and VMR2) through which projects acquire resources from the network are *jointly* constrained by network policies

- Where the most appropriate change methodology could be enacted within the organisation" (also refer Chapter 5).

If strategic management decided to pursue the creation of a partner network, then it will have to ensure that the mission, vision and strategic objectives of the company are in harmony with the intended mission of the network of enterprises and those of future partners in this alliance. Therefore, this decision centre works in close co-ordination with the corresponding decision centres of other partners. Freidheim (1998, p113) argues that decisions undertaken at this stage must be based on the company's understanding of the partners' objectives and an assessment of whether these objectives are consistent with the company's own.

To ensure that this realignment is in the best interests of the company, this decision centre relies on information coming from strategic product and resource management. The product side provides the requisite knowledge and strategic information about target markets and competitors as may be presented through market analysis reports. This information is weighed against the information provided by the resource side with regards to the capabilities of the company (in terms of its human, capital and machine resources) to meet the demands from the market. Mission realignment and negotiations with partners is constrained by the ability of the entire company to meet and adjust to the opportunities, the company's own overall strategy (refer Chapter 4), and the company's existing agreements with other existing partners. For a given 'business model' (the way of doing business) and the partner company's assumed role in it each partner may conduct a SWOT (Strengths, Weaknesses, Opportunities and Threats) analysis to ascertain what the future position of the company would be in the given business arrangement. Importantly, it is not very useful to conduct a SWOT analysis before identifying a potential business model, because the result of the analysis can potentially be very different from business model to business model.

This decision centre provides decision frameworks to three other decision centres. Within the company, it provides the guiding policies by which capability assessment and adjustment (1.3) will be considered, as well as to strategic product management (1.2) in order to finalize a product strategy consistent with the selected business model. Importantly, this decision centre also provides the framework for the planning of network (1.1) by establishing the jointly agreed upon guiding policies for planning the network operations and processes – in effect, establishing the autonomous domain of the network. This provides a framework for the network's strategic management, and may be maintained in a co-operation agreement. The agreement develops with time: in its initial form it mandates network management to develop a master plan for the network, but after it is developed and approved, it becomes a legal document guiding the co-operation of partners in the network.

Overall, the company's mission realignment, the new vision and objectives and plans are summarised in a Business Plan (Chapter 9).

12.6.3.2 Strategic Product Decisions (PMP1) as Undertaken by Individual Partners

The objective of this decision centre is primarily to increase the understanding of the external environment through the analysis and provision of critical information on the market with regards to opportunities and competitors. Strategic product management supports the strategic business decision process through the identification and assessment of the opportunities afforded through a strategic alliance and their potential impact on the company, and in general, the suitability of the business model for the exploitation of these market opportunities. Part of the task is to keep track of developments within and beyond the present markets – any developments that have potential impact on company strategy. To preserve the company's dynamism, up-to-date market information must always be available.

Strategic product management will therefore rely on the constraints and decision variables (1.2) provided by the realignment of mission and objectives. I.e., strategic product decisions can only be undertaken and pursued if they are in conformance with the mission, vision and objectives of the company. However, if product management is unable to work within the constraints of the present decision framework, change proposals must be tabled and either the company's strategy, mission and objectives have to be adjusted, or product management's constraints and limitations on its decision variables need to be relaxed.

Strategic product management also provides information about projected necessary capabilities for alternative product strategies. Capability assessment and adjustment (PMR1) uses this information to assess the company's own resources and to determine what adjustments might be needed for each alternative. Strategic product management also relies on information coming from the *network's* scope planning and strategic product management (NMP1) to ensure that there is co-ordination between the network and the company in terms of product decisions that might affect the company's participation in the network.

12.6.3.3 Capability Assessment and Adjustment (PMR1) as Undertaken by Individual Partners

The primary objective of this decision centre is to evaluate the company's status in terms of the adequacy and quality of its strategic resources (human, machine, financial) and to make appropriate adjustments where needed if and when the company pursues a strategic alliance. It relies on various critical information flows such as feasibility study results. In capability assessment, the company's resources are analysed in terms of potential product strategies, hence, there is information flow coming from the decision centre for product strategy making (PMP1). The result of this analysis is passed on to mission

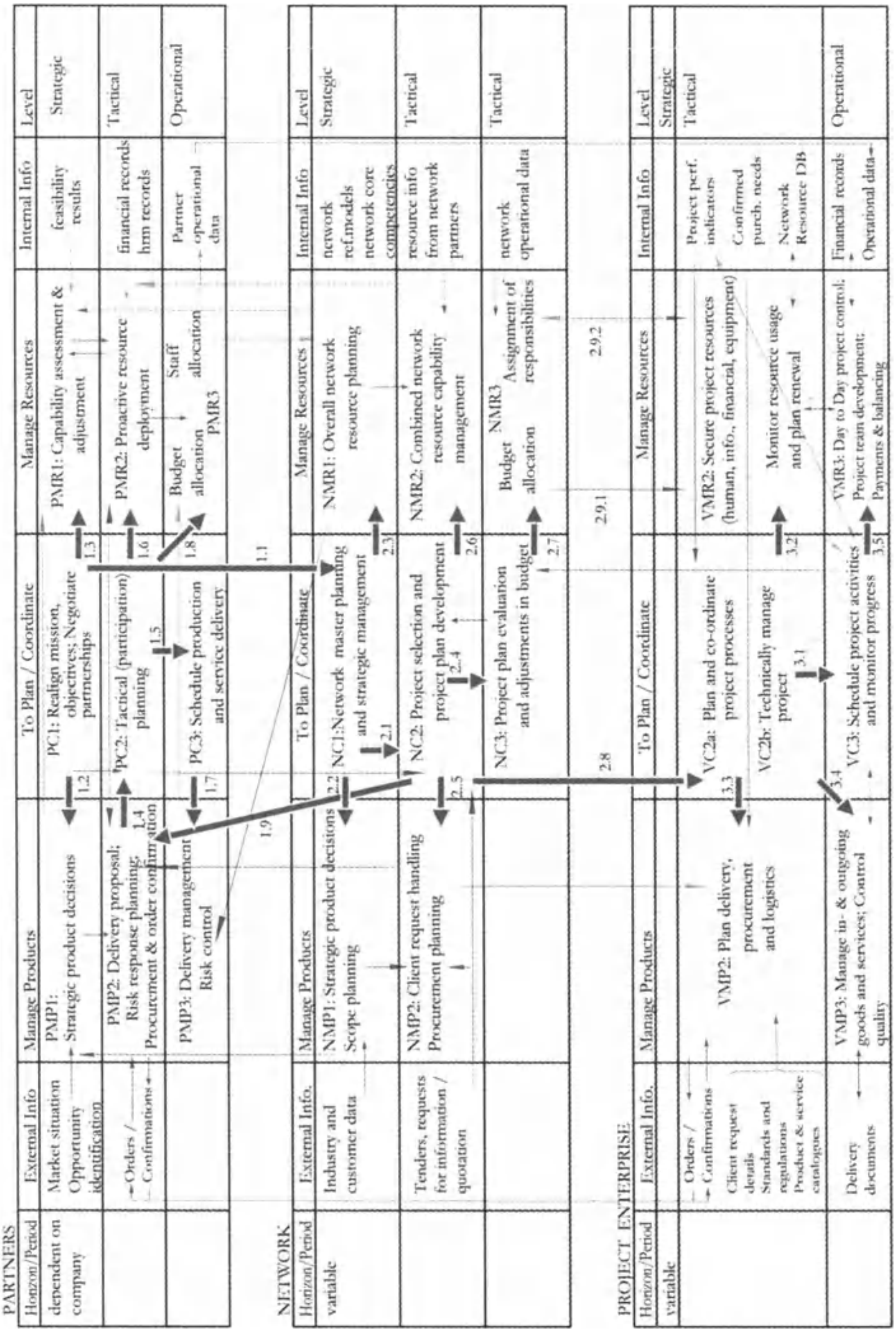


Fig. 12.8. Main decision flows between partners, network and virtual project enterprises

realignment (PC1) so that the adopted changes in mission and objectives will be consistent with what the company is capable of.
The decision framework of this Decision Centre (DC) (1.3) determines the actual resource adjustments to be made for the company to enter the network. Once the partner enters the network, this DC uses aggregate information and evaluates significant events coming from the tactical level to plan at the strategic level any necessary adjustments to the company's resource strategy (proactive resource deployment).
Because the resources include technology, it is the responsibility of this DC to monitor advancements in technology with the primary aim of identifying key opportunities of technological growth. It then sets about planning for the desired new level of preparation for the company's participation in the network. This includes securing required capital and funding, determining the quality that must be maintained by the human and machine resources, ensuring the required levels of commitment, and drawing development plans for human and technological resources accordingly (Uppington , 1998).
Similarly, human resource management not only has to plan for the needs of the network as defined at a given moment in time, but also needs to continuously monitor and strategically plan for any upcoming changes. This includes strategic decisions regarding staffing and staff development (such as providing career paths), as well as strategic knowledge management decisions, to ensure that corporate knowledge is shared, preserved and accumulated.
Last but not least the strategic management of financial resources includes financial planning, contingency planning, risk evaluation, and advice to overall strategic management regarding the financial arrangements of the proposed business model, as well as continuous monitoring and reporting of the network's performance. As with the other areas of concern, so in financial strategy making, anticipation of changes, and identification of significant events that need attention are within the portfolio of this management task.
The above development plans are not independent of the master planning of the network – strategic capability development must balance the internal capability plans of the company (as an independent entity), and the plans developed by overall network resource planning (NMR1) and proposed to the company (as a member of the network). Capability development in the company would use the enterprise architecture principles as described in Chapters 6, 11, 13, 16, the present Chapter, as well as Chapters 15, 17 and 18. Organisationally, partners may elect to delegate members of their capability development team to network resource planning, ensuring direct participation[20].

[20] small Note that depending on the circumstances this team might be a subset of the network planning team, or a separate group responding to the network planing team.

12.6.3.4 Tactical Planning (PC2) as Undertaken by Individual Partners

Tactical planning by partners is a task that concentrates on determining the participation of the company in any VEs that are currently in the planning-, or in the operational stage. It is assumed that the company has been pre-qualified to participate in the network and its virtual project enterprises. The nature of the project enterprises in question (e.g. bidding, construction, engineering or maintenance) will dictate the planning task that must be done for the company to provide to the virtual enterprises the competency for which it has qualified. It is for this reason that this decision centre's decision framework comes from the *product* side (1.4) – whatever medium term production and service objectives exist need to be taken into account, so as to be able to maximise the need for the company's products and services.

The primary tasks of Tactical Planning are to update the company's medium term plans, as well as to define the production and service objectives that the company should attain in order to achieve optimal success through the network. This decision centre's aim, therefore, is to determine attainable objectives for resource deployment, thus ensuring the company's participation in Virtual Project Enterprises being created and operated. The product of this decision centre is a readjusted overall schedule reflecting the company's available resources, current commitments, resource capabilities, and maintenance plans and schedules. The DC develops this schedule with due consideration of production and service objectives / constraints (1.4).

This DC qualitatively assesses what resources are necessary (people, equipment, materials and money) and determines what quantities of each should be used to perform the project activities with which the company is tasked; the success of the network depends largely upon the ability of the partners to plan, schedule and mobilise the necessary resources. Lorange and Roos assert that the success of this decision-making is the test of whether the co-operative alliance will have a chance to succeed (Lorange and Roos , 1992).

It is interesting to note that the decision framework (1.4) for the participation planning DC is not coming directly from the DC that determined the mission, vision and objectives of the company, but is indirectly determined (through PMP2) by the network (NC2). The review of literature shows that the company might be better off giving up part of its autonomy in this regard. This because the company's mission, vision and objectives would already have been aligned with other network partners – albeit autonomy is given to the network only for some management decisions and within the ambit of the realigned mission and objectives of the partners. Indirectly therefore, this participation planning is still being influenced by the company's mission and objectives. While the decision objectives and constraints come from the partner's product side[21] (1.4), the decision centre (PMP2) proposes the scope of the company's

[21] after all, on the tactical level the company relies on orders

involvement based on objectives, constraints and variables provided by the network (1.9).
The above statement has cultural consequences as well, because through giving up some autonomy, the hierarchy between people is changed [22].

12.6.3.5 Proactive Resource Deployment (PMR2) as Undertaken by Individual Partners

This management task (PMR2) is focused on the actual deployment of the resources to satisfy the combined schedules of network-related (and other) production and service plans of the company. This deployment has to be negotiated with tactical planning (PC2), hence the information flow between the two. Resource deployment relies on resource constraints and on business processes policies dictated by the strategic direction plans of individual partners (1.6) It is assumed that resource deployment plans are developed so as to support the partners' participation in projects of the network. For each partner, this resource deployment planning would use the partner's product-related decisions concerning delivery commitments, since the primary task of resource deployment is to make sure that appropriate manpower and machine resources are negotiated and apportioned to achieve product or service delivery. Resource deployment planning also gets from-, and provides feedback to the combined network resource capability management (NMR2). This information flow relies on the knowledge of jointly agreed upon tools and processes, which the company is expected to use to perform its assigned tasks. Each partner's resource deployment oversees internal cost estimation as well PMI (1996). Cost estimating on the partner level includes identifying and considering alternative costing solutions, to which tactical participation planning can refer for final selection.
Resource deployment takes care of staff allocation where the human resources needed are assigned to work on the project, with the decisions constrained by the company's own organisational policies regarding staffing.
This resource deployment task being on the tactical level is concerned only with aggregate resource commitments, i.e. with the estimated quality and quantity of resources to be allocated to planned projects for longer time intervals commensurable with project duration, or predicted intervals within the project plan as communicated to the partner through requests from the network (1.9). Once resources are committed to the network, individual detailed tasking and schedules will be developed by the respectively created project enterprise.

[22] refer the discussion of the allocation of decisional responsibilities to individuals and groups of individuals in Chapter 16(Organisational design)

12.6.3.6 Delivery Proposal, Risk Response Planning and Procurement (PMP2) as Undertaken by Individual Partners

This management task (PMP2) is crucial to the company's performance in the network as it is the 'contact point' between the network and the company. This DC gets its decision framework (1.9) from the network's project plan development (NC2). The objective of this DC is to lead the tactical level preparations for the company's participation in project enterprises, through:

- the development of proposals for products or services needed by project(s) in which the partner participates,
- risk response development and scope adjustment relative to the requested products or services, and
- procurement.

This DC is constrained by the pre-defined, jointly agreed-upon policies and procedures of the network and must therefore plan its activities within the do main defined by the network with due respect to the company's own strategic product decisions at the higher level (information flow from PMP1). This DC carries out its task with the assumption that the company has been pre-qualified to participate in the network's project enterprises. It is therefore governed by the policies and procedures of the network and the nature of the project enterprises requesting the partner's products, services or resources. Initially, the network defines the target objectives of the partners based on their roles in the VE and may outline performance and quality criteria to ensure consistency between all participating partners.
This decision centre is responsible for the development of the overall *delivery plan* that the company must follow, constrained by the time frame given by the network and based on the contract between the network and the client. Delivery plan development therefore negotiates with the company's overall planning system (PC2) in order to be able to make promises (i.e. propose a delivery plan for the requested services or products). The output of delivery plan development is the company's slice of the project plan that includes supporting details such as resource requirements by time periods or alternative schedules and a schedule management plan where decisions on how changes to the schedule are managed (PMI , 1996, p32).
Scope adjustment at the partner level is concerned with scope verification where there is a formal acceptance of the project scope as defined by the decision framework from the network (1.9) and corresponding adjustments are planned for. This involves product analysis, cost/benefit analysis and identification of alternatives and possible counter-proposals.
Risk response development is defined by the Project management Institute (PMI) as "defining enhancement steps for opportunities and responses to threats" (PMI , 1996, p32). Risk development starts from the identification and quantification of the risks most likely to affect the company's role in the

project. To develop contingency plans and an appropriate risk management plan, the DC relies on information from affected areas such as cost and duration estimates, proactive resource deployment as well as procurement plans from the network. Risk management plans outline the procedures used to manage risk throughout the company's involvement in the project. Contingency plans are predefined steps that must be followed should an identified risk event occur. According to the PMI (Ibid.), contingency plans may be part of risk management plans or may even be integrated into scope management or quality management plans.
Note: naturally, the plan to participate is a proposed project of the network by delivering certain types of goods and services is not a 'confirmation of order' – orders for specific products and services will be issued by the project enterprise. Thus, this DC is also responsible for receiving orders from projects and confirms these as necessary (and negotiates details with the project if necessary). At the same time, this DC initiates all procurements that the partner needs to satisfy orders received from projects.
While delivery proposal, risk response planning and procurement have the same (or similar time) horizons these activities (from a given project's point of view) follow one another with some delay. In practice, the tasks of this management role may be distributed among several people.

12.6.3.7 Operational Management Delivery and Risk Control (PMP3) and Scheduling (PC3) as Undertaken by Individual Partners

These decision centres are conventional operational management tasks controlling the production of goods and the delivery of services. Some of these deliveries are for projects of the network, and some to other customers. The level of priority given to orders coming from the network's projects is determined by a negotiated network policy. Typically, there would be other types of priorities as well, which individual partners use when developing schedules to satisfy orders.
Operational scheduling (PC3) is concerned with the conventional daily- or weekly task assignments to individuals and machinery, including the scheduling of logistic services provided by external contractors. Normally, project schedules (developed by projects – VC3) may take delivery dates promised by partners as given (with slack times built-in for contingencies). However, the use of advanced scheduling software by all partners may allow the partners and the projects to use distributed scheduling for further optimisation.
The operational management of products (PMP3) oversees the incoming and outgoing products and services for the partner. On the incoming side, this DC manages the incoming products (materials, tools, parts, subsystems) and services necessary for the partner to fulfil its obligations to projects. On the outgoing side, the DC manages the delivery of products and services, including

final quality control and dispatching of deliveries[23]. When schedules become for any reason impossible to maintain, a suitable response is expected, so that projects do not become affected. This additional task is called 'risk control' in this model.
There are two questions to be answered by risk control:

- How is it possible to maintain delivery schedules without adverse effects on projects? Consequently, this DC would have to negotiate with the project scheduling concerned (VC3);
- What types of changes are acceptable from the individual partner's point of view? To answer this question, the company's individual plans and schedules must be taken into account (so as not to adversely affect other projects or customers).

Risk control must consider the policies developed on the tactical level (refer to 'risk response development' – PMP2). However, these policies are only guiding what types of activities are admissible to negotiate risks. The actual objectives come from the negotiation between the partner and the projects affected about possible changes to the delivery schedules.
Risk control involves identifying, quantifying and responding to risk events as needed, and therefore it relies on the operational level and real time level activities on the resource management side to attempt an appropriate corrective action. Risk control and resource management cannot, however, come to final agreement regarding the solution to the problem; the solution needs to be meshed into the overall operational schedule by the respective authority (PC3).
Risk control also provides feedback to the tactical level for risk response development so that the risk management plans are updated, enhancing the dynamism of the company. Risk control is constrained by the predefined budget from the coordinating authority at the tactical level, as well as the terms of the contract the company has with the network/client.
Sometimes the only plausible solution to the risk event is the procurement of additional tools and materials or engaging external service providers to satisfy the partner's commitments.

12.6.3.8 Budget Allocation / Staff Allocation (PMR3) as Undertaken by Individual Partners

Notice that Budget and Staff Allocation is not directly guided by tactical level resource management, or by operational scheduling. Instead, it is based on the decisions taken by the tactical planning authority (PC2) of the partner. This is to ensure that the allocation is directly based on the company's

[23] Note that project enterprises also have their own delivery management – refer Section 12.6.5.3

commitments[24]. The DC gets its framework (1.8) from participation planning and is constrained by the company's financial resources, as well as its human and machine resources. Detailed information from the tactical level resource management (proactive resource deployment) is taken into account in order to consider tactical resource management decisions; these however, are less constraining then the actual objectives committed to by individual planning. Thus, budget estimates are worked out and disseminated to all involved actors within the individual partner organisation.
While the decision framework (1.8) of PMR3 is determined by a higher level, the actual requests for staff and other resources comes from operational scheduling, because the decision framework is only *guiding* the operation of this DC.
The granularity of resource allocation is different between *partner* resource management and *project* resource management. Project resource management requires from the partner a type of individual with given attributes, while partner resource management allocates actual individuals to the tasks. The selection of the individual would be typically done by a functional line manager of the partner.
In knowledge intensive projects there may be a need for the individual allocation of human resources directly to projects, in which case project planning (NC2) must in addition co-ordinate directly with individual company planning (PC2). For example, in consulting services it is customary to contractually define the participation of certain key personnel. The situation is similar for unique resources, such as un-substitutable high value manufacturing or other equipment.

12.6.4 The Network

The main purpose of the extended enterprise network is "to prepare and manage the lifecycle of VEs and to prepare them for the implementation of product lifecycles" (Globeman21 , 1999). The purpose of this section is to discuss management tasks for the network entity and show how decision frameworks are passed between the partners, the network, and the project enterprise that the network creates.

12.6.4.1 Master Planning and Network Strategic Management (NC1)

Partners initially set up the network's strategic management in form of the Network Planning Team. When the network starts operating, partners may decide to appoint new network management, or transform the planning team

[24] implementors of this reference model might consider moving the budget allocation DC to the tactical level, or to a longer horizon operational level (splitting PMR3 into two), with staff allocation remaining on the operational level

into a permanent network management team (or allocate the task to one individual network manager).

Master Planning the Network: Initially it is the network master planning team which acts as the top-level authority of the network and is responsible for the overall planning and co-ordination of the network - covering the lifecycle phases from the definition of requirements down to implementation (building). The primary business objective in this *stage* is to oversee the conceptualisation, development and implementation of the master plan for network operations and make appropriate decisions (e.g. development of project plans) based on feedback from the product and resource side, in order to reflect the needs of the environment in which the network exists. This environmental information flow may take the form of market analysis reports on future market opportunities for the network's class of products. From the resource side, the network looks at the research and development plan, as well as the overall resources of the network (e.g. core competencies) for planning purposes.

Consequently, the network aims to establish appropriate processes and organisational design to shorten the response time to current opportunities and ensure that the business network is responsive to future market opportunities. Decisions here relate to the planning and strategies of the network, in accordance to the jointly agreed upon business objectives of the partner members (1.1). Hence, this decision centre operates within the decision framework provided by the partners. Its autonomy is defined through negotiations between the partners who will have developed the network's mission, vision and objectives and who are responsible for the Identification and Concept phases of the network.

Some of the decisions that may be considered at the network requirements definition phase involve the setup of the overall management structure of the network and corresponding lower level business processes and policies. Depending on partner intentions, network management may have the freedom to select the types of products/services that the network will cater for, however, the class of products for which the network is created may already have been defined by the partners – as the case may be. I.e., a network may be created to share resources, for whatever purpose they are fit, or to pool resources to be able to satisfy certain customer needs (e.g. to design and build a given class of products, or to provide certain types of services). Note that the considerations of *possible* network products and services (as proposed by Network Scope Planning – NMP1) and the *actual* decision to select from the possibilities (done by Network Master Planning) have been separated. The reason for this separation is that Network Master Planning must consider the proposals made by Network Resource Planning (NMR1) and has to strike a balance between these and the possibilities offered by product- and resource planning.

Network Master Planning also considers the standards and methods to be followed by partners, which includes the design of processes to be followed by the network. Decisions regarding detailed policies for future partner selec-

tion also fall under this decision centre's domain. While partners will have developed overall policies in this regard, master planning will inevitably find it necessary to develop more detailed policics to make sure that all network processes have requisite policies to follow. Thus, overall policies as negotiated by partners will be decomposed or detailed into process-specific policies.

As a summary, this decision centre consolidates the results of overall network resource planning and of scope planning into one consistent and coherent plan to guide the network's design.

The tasks of this DC are extensive, and include (reference to chapters that treat relevant methods is given in brackets):

- Requirements specification of the Network (Chapters 11, 13 and 16),
- The architectural design of the network (Chapter 15).

Depending on the nature of the network, Network Master Planning may have to include the *master planning of typical project enterprises as well* (see also the discussion of this matter in Section 12.6.2) – except in case of loose networks, where insisting on common project practices may not be viable. In case the network is not a loose alliance of companies, the master planing of project enterprises includes the same types of tasks as those listed for the network above.

The master planning effort for the network and for project enterprises would typically use reference models (such as the model discussed in the present Section 12.6), and would make necessary adjustments and adaptations for the particular network. Also, the network planning team should decide on what general design principles and patterns to use in order to improve the quality of their design. Section 12.4 has discussed such principles.

Relevant reference models for the design of commonly adopted processes for project enterprises include:

- ISO 15288 (Systems life-cycle processes) (ISO/TC184/SC5/WG1 , 2000),
- The Project Management Body of Knowledge (PMBOK),
- Partner Interface Processes of Rosettanet for electronic business to business transactions (Rosettanet , 2003);
- Product data exchange standards (e.g. STEP (ISO10303) (ISO , 1994)),
- Interoperability maturity models (Levels of Information Systems Interoperability – LISI, in (C4ISR , 1998)),
- Quality standards, such as EFQM (EFQM , 2000) and ISO 9000:2000 (ISO , 2000).

as well as a number of other relevant industry standards. The network planning team must conduct a review of relevant standards to be adopted and applied in the master plan.

Once the Network Master Plan is complete, the planning team must present the result to the partner companies, together with plans (budget estimates, time required, risk) for the detailed design and implementation of the network.

Partner companies must select from the options (or request changes to the master plan or to the implementation plans) and decide on the selected option. At the same time, partners decide on the possible change in the personnel in network management, and give the new management team a new mandate (through its decision framework 1.1) with objectives, budget, time frame and any decision variables and constraints that define the authority of network management in its next stage of operation.

Note that the detailed design of the network and of its infrastructure and the implementation of this design is not in the scope of this DC; this will be in the scope of overall network resource planning (NMR1). Possibly some members of the master planning team will be moved to resource planning from this time on.

Management of Network Implementation: Network management will develop the mandate (2.3) for network management to carry out the detailed design and implementation project of the network, providing it with objectives, budget, time-line, decision variables and constraints for the performance of this task. Network resource planning will report to network management as necessary and network management will make any decisions and approve changes as long as the result is within its defined authority.

Strategic Management of Network Operations: When network implementation is complete, network management will report to partners and propose the start of network operation. As part of the implementation project, network resource planning must have carried out all individual and integration testing, and the network is ready to receive customer requests.

As part of the implementation project, all decision centres of the network have management personnel allocated to it, and decision frameworks are set up to define their authorities. Therefore, network management can assume its usual strategic role, of continuously monitoring and guiding the network, assessing change proposals to the network's product and resource strategies, or initiating the development thereof, and as a result to update the decision frameworks of all decision centres that operate directly below it (NMP1, NMR1, NC2).

This DC sets objectives for project plan developers (NC2), provide guidelines based on which project plan developers may autonomously prioritise client requests, and policies that guide project plan development processes.

While network strategic management operates in relative autonomy, it must provide performance reports to partners for them to be able to assure themselves that the network keeps operating within its mandated boundaries. In case partners decide to change the mandate of this DC or to change the constraints or decision variables that it is allowed to manipulate, strategic network management will have to propagate the consequences to its subordinate DCs listed above.

12.6.4.2 Strategic Product Decisions and Scope Planning (NMP1) by the Network

Scope planning involves the development of a scope statement, which will serve as the basis for future project decisions, specifically focusing on criteria used to determine whether a project can be expected to be completed successfully (PMI , 1996) by the network. The objectives of this decision centre are:

- To facilitate the extended enterprise's understanding of the market's changing directions and trends, as well as customer requirements, market-perceived value of products and services provided by the network and the understanding of the competition;
- To maintain an understanding of the external environment and support NC1 in the process of taking strategic business decisions.

In the initial (setup) stage of the network, scope planning is necessary to give the partners a feel for the possible arrangements and target markets the network can operate in, with due respect to the set of core competencies within the network. The outcome of scope planning forms part of the basis for the agreement between the different partners by the identification of the demands the network anticipates to fulfil, as well as expectations for future market development.
The crucial decisions include:

- the composition of a preliminary project list for the network, and
- the projection of the expected future situation of the network.

This decision centre therefore relies on external information such as industry and customer data as well as market analysis reports. These reports analyse the opportunities and threats and assess them against the network's strengths and weaknesses. The DC closely coordinates with the decision centre on overall network resource planning (NMR1). It exchanges information with network resource planning about the impact of customer expectations and market / product characteristics on the network's resources (inclusive of the core competencies). This DC provides analysis reports to the DC that coordinates the master planning of network operations (NC1). It also provides feedback at the tactical level for the partners (PMP2) so that whatever scope adjustments the partners will have to make at their own level, will be done in accordance to the planned scope of the network.
During the operation of the network this DC continues to develop possible new product strategies, proactively seeking viable variants. However, final product decisions are made in accordance with the overall network strategy as decided by NC1. This DC develops its recommendations in line with policies on marketing and sales strategies of the network. In this particular design, strategic product planning has indirect control over tactical product decisions. However, NC1 and NC2 make their decisions on the basis of this DC's recommendations. Thus, while product strategy forms one of the two pillars of the

network's strategic direction and determines the future ability to undertake projects (or in general to form VEs), the handling of actual customer requests is not interfered with.

12.6.4.3 Overall Network Resource Planning (NMR1)

The overall business objectives of this decision centre are to initially implement all necessary resources for the network and, during the operation of the network, conduct continuous planning and development of network resources. The task list of this DC includes:

- Implement methods and build or commission the deployment of tools to be shared by network partners. This includes all shared tools, such as those supporting business transactions, communication and information sharing (collaboration tools, shared databases and repositories, etc), planning and scheduling, project management, and any other necessary management information system applications. Furthermore, any applications that are necessary for the interoperation of partners need to be installed as well. If the network adopts common tools for mission delivery, then Network Resource Planning needs to help partners to install the correct versions of tools, and supervise the interoperability / integration testing of these;
- For some business processes that need to be elaborated on the detailed design level, Network Resource Planning develops these detailed descriptions and / or implements them using 'workflows'. For those business processes (or parts of business processes) that are human-implemented, suitable training and documentation must be developed, training plans drawn up and the training must be secured for the partners' relevant human resources (possibly using external assistance);
- Continuous monitoring of resource cost and performance in the network, and making timely proposals to the network's strategic management for any change in processes or technology, training, or necessary upgrade;
- Continuous monitoring of technology and general concepts (including human resource concepts) and producing technology forecasts;

This decision centre coordinates closely with the product side strategic level of the network by the provision of feedback regarding the feasibility of possible product strategies as well as possible modes of operation. Initially, in the network setup stage, this is to support the master planning of the network. During network operation, this DC continues the same decision support role, but at the same time develops initiatives for network resource development.
Perhaps the most critical decisions in this centre are the choice of the information and communication technology (ICT) *integrating infrastructure* and the development or deployment of the *project management tools and procedures* that future VEs draw upon (and thus must be deployed at partner companies). This DC decides on the requisite partner competency for all ICT and

provides feedback directly to the partner's strategic level decision centre on *capability assessment and adjustment* (PMR1) so that appropriate action can be taken at the partner side. It relies on internal information such as network reference models, and on the knowledge of core competencies in the network. The DC uses the architectural design of the network as delivered in the Master Plan for the detailed design and implementation of the rest of the network, and carries out its detailed design and the implementation (refer Chapters 16, 17 and 18 for relevant methods). It receives its mandate from the master plan for network operations (2.3) and is constrained by the availability of the combined network resources (human, financial and machine), the overall business strategies of the network, and the business rules for managing the network. Given that the detailed design and implementation task is resource intensive, the DC would rely on existing resources of partners and external contractors to perform its task. However, if the detailed design is carried out by external contractors this DC must have the competency to evaluate the results and to maintain an up-to-date documentation.

Once the network is operational, this DC has an ongoing strategic responsibility for network resource development. Therefore, continuous development of requirements and conceptualisations are necessary for human resource management (job description, procedures for each role) as well as for all technical and any other resources of the network. These are then passed on to the strategic resource management (PMR1) of partners so that improvements in network participation can be planned[25]. In particular, requirements and conceptualisations are made for the continuous development of ICT tools and integrating infrastructure.

12.6.4.4 Project Selection and Project Plan Development (NC2)

This DC is responsible for tactical level decisions to create virtual project enterprises and supervise their operations. It operates observing its decision framework (2.1) determined by the network's strategic management. Thus, project plan development gets its overall business objectives from the network, and based on these responds to existing or anticipated market demand through initiating project enterprises.

This decision centre is essentially network management's decision on how to respond to market stimuli, be they a market request, an opportunity, a

[25] This direct feedback between the Network and partner strategic resource management is an *informative* channel of communication, since network resource management does not have the authority to set objectives for the partners. However, partner strategic management would be able to mandate policies and procedures (consistent with the agreed charter of the network) that govern how this type of feedback needs to be handled. While direct link between the corresponding experts of the network and of partners is crucial, partners must have an *autonomous* strategic resource management role for themselves – after all participating in the network is only one facet of their business activity

business requirement or a problem. This requires information on the product or service proposed and on the availability of network resources to develop an adequate response. The decisions are made here in the light of the network's strategic plan (that is part of 2.1).
Given that project plan development is usually a response to market demand, the task must be performed without delay. Therefore, this decision centre must use a previously developed project reference model (or one out of a selection of reference models) to expedite the task. Thus, project plans are not created from scratch, but through parametric adjustments to existing 'blueprint' plans, or using a building block approach (assembling a plan from tested components).
The DC selects partners to participate in the planned VE based on their core competencies and constrained by the criteria and standards that must be maintained from the overall master plan and the contractual provisions from the client. Thus, project plan development communicates directly with partner participation planning (PC2). It also relies on internal information, such as historical data summarising the results and partner performance of previous projects (from NC3). The output (2.8) of this decision centre includes the project charter (including financial and time constraints) and composition (the non-negotiable part of the list of participants), the assignment of the project manager, the identified resource contribution from the partners (including project budget) as well as any constraints that the project manager must observe.
An important task for the network's project plan development is to consider *all* projects (including actual planned and future potential ones) and use aggregate life-cycle costing for these. For example, it is possible to optimise the total life-cycle cost of the product (from the customer's perspective) and to calculate total life-cycle profits aggregated over all projects (from the network's perspective). Thus, while network resource planning (NMR2) provides this DC with costing alternatives for each project, the selection from the available alternatives needs to consider the above composite figures over a longer period of time and over multiple projects.

12.6.4.5 Client request Handling / Procurement Planning (NMP2)

This decision centre is the contact point of the Network to the external environment (clients and suppliers). Its primary business objective is to handle all transactions with the clients and to support the network's understanding of customer's needs. To be more specific, this decision centre is tasked to:

- Identify customers' requirements, customers' profile, upcoming tenders as well as issue requests for proposals to potential suppliers;
- Manage the contractual details in customer contact (for bidding and service / product delivery projects);

- Initiate and set up collaboration with suppliers, and in general manage supplier relationships (e.g. preferred supplier schemes);
- Develop aggregate purchase plans / forecasts, and secure supplier contracts in support of these.

One of the main responsibilities is to ensure that the process of identifying, contacting and acquiring new customers is effective and that new requests for tenders are efficiently dealt with and evaluated. Once a VE project charter (2.4) is in place, NC2 determines procurement planning objectives and constraints for this DC. NMP2 uses feedback coming from the combined network resource capability management (NMR2) to determine services that will have to be secured from outside the network. Note that this is not an actual purchasing task, only the planning and securing of suppliers to be used by purchasing (as will be done by projects). Procurement planning does not necessarily mean that purchasing has to be centralised. There must be a plan on how materials and services are procured and this must be incorporated into the VE charter to guide the project enterprise accordingly. However, the network may prefer a centralised purchasing function for certain materials, components or services that are sensitive in quality or otherwise not generally available on the market.

12.6.4.6 Combined Network Resource Capability Management (NMR2)

This centre is concerned with tactical decisions concerning the pool of network resources available for use by project enterprises. The primary objective here is to ensure that network resources are available where needed and when needed. At the human resource side, vital decisions in this centre include the delegation of roles and responsibility assignments, and the establishment of reporting relationships. This information flow goes back to the project plan development (NC2) for approval and consolidation. Human resource-side decisions are constrained by the availability and competencies of human resources made available by the partners. For the technical resources (including ICT tools), the requirements and guidelines from the strategic level (overall network resource planning) are translated into architectural and detailed designs and the VE partners implement and test the above-mentioned tools accordingly. The results are amalgamated and managed as a combined pool of network technical resources.
This decision centre develops feasible resource allocation alternatives to planned projects and gives this as decision support information to project plan development (NC2). As for the financial resources, decisions of this centre revolve around cost estimations of the resources needed to complete the VE project activities. Furthermore, plans are developed for managing possible variations in costs. This includes the provision of costing alternatives, from which the decision centre on project plan development (NC2) may choose. The

PMI (1996) calls this 'life-cycle costing'. In this case, a more appropriate term would be 'VE life-cycle costing'.

12.6.4.7 Project Plan Evaluation and Adjustments in Budget (NC3)

The primary business objective of this decision centre is to ensure that the VE's performance is being monitored and is in accordance with the tactical decisions passed on by the network to the VE (in the project charter, 2.8). This objective may be attained through cost control and performance checks, based on pre-defined policies, performance criteria, and the budget given to the project enterprise. Cost control involves monitoring the VE cost performance for variances and counter-checking to ensure that inappropriate changes are excluded from the cost baseline drawn by the network for the VE (PMI , 1996). This decision centre constantly monitors feedback from the VE project enterprise for variances. As a result, decisions on revisions for cost estimates, budget updates and corrective action can be made by the network and can be disseminated accordingly to the involved partners. Significantly, knowledge management is critical in this decision centre, so that the causes of variances are discovered and corrective actions can be taken, as well as all the other 'lessons learned' (knowledge) from the project may be collated and documented for use in future projects of a similar nature. These aggregate reports are a feedback to project plan development (NC2).
This decision centre is constrained by the performance criteria as well as the other guiding project policies and processes from the decision centre on project plan development (NC2). On the other hand, it provides the decision framework on the tactical level to the resource side on budget allocation and assignment of authorities.

12.6.4.8 Budget Allocation and Assignment of Responsibilities (NMR3)

This decision centre is responsible for allocating the actual financial and human resources of the network to the project (note that NMR2 accomplished only *planned* aggregate allocations). The task is to coordinate the allocation of the essential resources (financial, human, machine) to projects. This is done on the basis of the selected alternatives of project resource allocation plans[26]. Allocation is made to individual work items as demanded by each project and resource usage is monitored in order to establish the cost baseline to measure the project performance. The business objective is attained by apportioning the financial resources based on the negotiated budget and the delegation of suitable personnel to the defined key roles. The selection from the pool

[26] note that these plans will have been developed by combined network resource capability management (NMR2) and approved by project plan development (NC2)

of available network resources must take into account optimum criteria for overall resource usage in the network, network policies for resource allocation, as well as project constraints and project risks. Thus this DC must negotiate and closely collaborate with the resource allocation side of projects (VMR2). For budget proposals and responsibility assignment, internal information is needed about the network's operation and the decisions are constrained by the policies on budget and human resource management previously adopted by the partners.

12.6.5 The Virtual Project Enterprise

The virtual project enterprise considered in this reference model is any joint undertaking that the network decides to form, through a strategic match of the core competencies of pre-qualified partners. Thus, it may be a virtual project enterprise for bidding, or a virtual project enterprise for engineering a product, or even a virtual enterprise for the maintenance of a product (in the latter case the virtual enterprise may operate for a long period of time, unlike a project). From Fig. 12.8 it can be seen that the virtual project enterprise gets its highest level decisions from the network (2.8, 2.91, 2.92). Hence, given the time frame of projects, the project enterprise starts to operate at the tactical level. It is given the autonomy to make decisions, but only within the decision frameworks provided to it by the network.
The decision centres that constitute the Virtual Project Enterprise shall now be discussed.

12.6.5.1 Plan and Co-ordinate Project Processes (VC2a) / Technically Manage Project (VC2b)

In larger projects, this role is usually divided into project management and technical management roles. In smaller projects, the two might be considered as one role, as long as competent human resource is available to fill both roles. Thus there are two management tasks:

- Make human role assessment and, relying on technical management advice, make decisions about the activities to be performed by the project, allocate human, technical and budgetary resources in a timely manner, monitor progress (define and monitor performance indicators and milestones), take corrective action as necessary, and negotiate with the network in case the project charter needs amendment or modification;
- Develop the process that will fulfil the project's mission[27], define and monitor the technical / quality characteristics of project activities as well as

[27] The level of granularity is defined by three requirements: a) activities that are likely to be performed by different people / organisations need to be defined so that events between these activities become observable by whoever is co-ordinating them, b) if a process has multiple activities performed by one person,

of their outcomes, decide on technical alternatives regarding processes to be used, as well as on what alternative outcomes are most suitable to fulfil project requirements, propose to the project manager any items that must be negotiated with the client and / or the network.

Manage progress (*project management role – VC2a*): This decision centre is responsible for the overall direction of the project, establishes the chain of command within the virtual enterprise, and allocates resources to project activities. Even though projects are set up on the basis of project reference models used throughout the network, project management must have sufficient authority to flexibly organise and re-organise the project team as the task requires. The constraints for this decision centre are the VE charter (the defined task, together network policies and standards to be maintained), terms from the client contract, predefined budget from the network as well as legislation and resource availability (Fig. 12.8). Project management is the primary point of contact with the network, it negotiates the terms of the assignment and reports to the network on project progress.

Make technical decisions (*technical management role – VC2b*): This decision centre gets its mandate from the project manager (VC2a), operates within the constraints of the project charter (2.8), and coordinates the project's product and resource management sides. The primary objective of this decision centre is to identify in detail what the VE must perform in its operation phase and how these tasks are going to be performed. The planning process covers requirements analysis[28] and design of the mission delivery processes (such as engineering, design, implementation, etc.) and the development of a detailed project plan. Process design is based on the co-ordination needs of the project, and on the knowledge of the *demonstrated* capabilities of available resources. This role also performs cost budgeting and approves tactical level resource plans, or a selection from alternatives is made (on the advice of tactical level resource planning – VMR2). Thus technical management presents resource management with the requirements and constraints (3.2) and resource management responds with possible alternatives. Technical management also supervises the planning of product delivery, procurement and quality assurance (VMP2).

While the project manager's role is primarily concerned with the main mission delivery process of the project and the co-ordination tasks between partners, the technical management role's scope includes all processes (mission delivery and logistics).

It is essential that potential project managers / technical managers are trained in the policies and procedures of the network (as defined in the network's

---

but events or information on the interfaces of these activities must be observable for co-ordination purposes, then the activities of the process must be spelled out, c) the level of detail should match the demonstrated capabilities and competencies of resources available for the execution of the process

[28] The requirements analysis starts on the basis of the project charter.

project reference models), because at the time a project enterprise is set up there is little time for training.

12.6.5.2 Secure Project Resources, Monitor Usage and Plan Renewal (VMR2)

This decision centre's objective is to secure and allocate all necessary project resources (human, technical, financial), as well as to proactively monitor resource usage, planning timely renewal or replacement as necessary. The role is performed under the guidance of top level project management (VC2). Resource planning is continuously performed during the project, according to the defined stages of its progression.

Secure project resources: Project management (technical management) provides this role with the description of the processes that need to be performed, and specifies the capacities and capabilities necessary, together with quality and timing criteria (such as project milestone dates). Based on the task descriptions (as part of 3.2), and known characteristics of potential resources, such as may be available by accessing a Network Resource Database (NRD), this DC estimates the duration of individual project activities, and taking into consideration of timing constraints defined in the project plan (received as part of 3.2) determines the amount of resources needed as well as approximately when they will be needed.

The objective of estimating the duration of project activities is to provide reasonable forecasts of work periods needed to complete high-level partner activities. Decisions at this stage are constrained by resource availability and the contractual requirements (imposed dates) from the client. Internal information flow may also come from historical data such as previous project files, project team knowledge (experiences from participating team members). External information from commercial 'duration estimation' databases are also useful (PMI , 1996).

This DC negotiates with the Network's tactical level resource management (NMR3) and develops resource assignment proposals for all processes in the project (including core and support activities). Depending on the objectives determined by project management (VC2) this DC could perform an optimisation[29] to propose resource usage that best satisfies project objectives. If the network is unable to provide resources for the project that meet the needs and project constraints, external resources are sought for.

The decision variables for this role would typically allow the DC to act in relative autonomy (e.g. in its negotiations with external suppliers), but the final acceptance of the project's resource selection is made by the project management role. Since in many cases resources can be pooled, the actual allocation of resources to tasks is done by activity scheduling (VC3). Note

[29] Such as optimisation for shortest time, lowest cost, highest profit, lowest risk, maximise the use of network resources, etc.

that this role is not responsible for raw material and component procurement – that is the task of another DC (VMP2).

Monitor resource usage and plan renewal: In receipt of aggregate performance indicators from the day-to-day management of the project (VMR3), as well as in response to any other significant events, this DC develops improvements to the project's resource plan, and endeavours to take corrective action before the problems noticed start to show effect on the progress of the project or on the quality of the outcome.

This management role of each project in the network may, for example, use and contribute to the maintenance of a Network Resource Database (of internal human and technical resources and of external suppliers) to collect and improve the information that will be used in the future to make resource planning decisions. At the latest, such information needs to be passed on to the Network in the project decommissioning stage. Note: such a Network Resource Database itself is an information *resource* of the network (and thus is expected to be described in the NRD itself).

12.6.5.3 Plan Service and Product Delivery, Incoming Goods & Services Procurement and Logistics (VMP2)

Plan Service and Product Delivery and Logistics: This decision centre is responsible for planning the delivery of the project's desired outcomes. Plans must include the specification of the quality, quantity, state[30] and time of planned deliveries (including deliveries internal to the project). This includes logistic planning, such as transport, storage, packing and unpacking, formation of batches, provision of energy, consumables, maintenance as well as environmental plan (human, technical, natural)[31].

The basis of the above planning is the detailed project plan. Depending on the competencies and the amount of human resources allocated to this DC (and to the project management role VC2), the detailed project plan may either be developed by this DC for the approval by project management, or (as it may happen in smaller projects) project management would develop the detailed project plan and ask this DC to work out additional details (such as logistics).

Plan Procurement of Components, Materials & Services: The objective of this activity is to manage whatever centralised purchasing is necessary[32], based on the needs determined by service and product delivery planning (above) – i.e. given the specification of the quality, quantity and timing of all needs. Centralised purchasing is advantageous in terms of quantity dis-

---

[30] the 'state' of a product includes the specification of the location, packaging, storage, etc. characteristics.

[31] ideally, internal deliveries in a project have the same or similar status of importance as external deliveries, given that the mission delivery process is carried out by the collaboration of multiple partners and external service providers

[32] since part of purchasing for the project would be done by partners

counts and direct project supervision; however, additional logistic tasks may be involved (which incur costs).
Note that the aim of procurement planning by the network (NMP2) was to predict the aggregate needs of the project enterprise for cost estimation (surveying) and budgeting. At the *project level* these aggregates must be decomposed, detailed and/or confirmed (by the project's technical processes) and the actual procurement plan must be based on these confirmed needs (refer to 'confirmed purchasing needs' in Fig. 12.8).
This DC is also responsible for quality planning and outlining quality management procedures that must be adopted by the project enterprise. Project quality management must deal with concerns on the management of the project enterprise, as well as the product of the project enterprise (PMI , 1996). The decision variable therefore, is the selection of quality standards relevant to the project and the means to satisfy them through the provision of operational definitions to guide the production activities based on the nature of the product concerned. This aspect of the decision centre gets internal information flow from the quality policies of the network and the VE charter. It also relies on external information such as the product description from the contract and national / international standards and regulations. It provides feedback to the planning process for consolidation.

12.6.5.4 Schedule Project Activities and Monitor Progress (VC3)

The objectives of this decision centre are to maintain the up-to-date schedule of the project and to monitor the progress / performance of the project enterprise. This decision centre operates within the ambit of the detailed project plan (as approved by technical management – VC2b). The project schedule includes the planned start and expected finish dates for each project activity[33]. This decision centre has the authority to control the schedule to ensure that the project enterprise's major deliverables are on time, working within the constraints of the approved resource allocation plans (thus the information flow from VMR2). At the same time, this DC receives information from the management of incoming and outgoing products and services, to ensure that only those activities are scheduled for which the preconditions have been satisfied. Also, scheduling receives up-to-date status information from the daily control of the project (VMR3), ensuring that only such activities are scheduled for which the planned resources are available at the desired times.
The PMI recommends the use of a schedule change control system that defines the procedures by which schedules may be changed along with the nec-

[33] 'Activity' here means the lowest level function that is to be controlled / monitored by management. I.e. given the specification of the desired outcomes of the activity there is a human or technical (or hybrid) resource that is capable of producing the desired output without management's intervention / co-ordination. For the allocated resource the activity may in turn be a complex process, however, the details of that process do not need to be visible to project management

essary approval levels. Critical decisions at this phase include the assessment of whether a schedule variance requires corrective action. Schedule variances are monitored through performance measurements / criteria as outlined by the VE planning process. This decision centre is tasked to provide schedule updates and to inform the appropriate stakeholders as needed. This decision centre provides aggregate performance indicators on the basis of operational data. This information is used by project management and procurement and delivery planning for future project activities[34].

12.6.5.5 Manage Incoming & Outgoing Goods and Services and Control Quality (VMP3)

The objective of this decision centre is to determine whether the outcomes of project adhere to the agreed upon-, as well as to the mandatory quality standards and to monitor all incoming and outgoing goods and services. Note that this DC not only monitors the quality of products and services but also the quality of information. The DC may make acceptance decisions, as well as process adjustment recommendations as necessary. This decision centre works under the direction of the project's technical management (VC2b), and maintains information contact with project scheduling (VC3). This management role receives performance criteria as well as the relevant standards that must be adhered to as set out in its decision framework (3.4) determined by technical management. External information flow (e.g. orders and their confirmation) is used to control the incoming and outgoing goods and services. Corrective action recommendations and analysis is passed back to scheduling and progress monitoring (VC3).

12.6.5.6 Day-to-Day Project Control, Project Team Development, Payments & Balancing (VMR3)

Day-to-day project control: this management role is responsible for the hands-on operational management of resources – ensuring that on the day-to-day basis tasks are allocated to people according to schedule and proposing any changes to be approved by project scheduling (VC3). This role manages both direct mission delivery activities as well as all maintenance tasks.
Project team development: operational control being in day-to-day contact with project personnel needs to ensure that project teams are harmoniously working together. This management role might for example use team-building activities with the intent to improve interpersonal relationships among key players and the provide any short training for participating members. Many factors influence the success of this management role, such as how

[34] It is assumed that detailed schedules are continuously developed and revised, therefore such aggregate performance data can be used to make timely correction to the detailed project plan and corrections to resource plans can be made accordingly

well the network culture has been adopted by partners, the strength of the integrating information and communication infrastructure to facilitate communications, the level of maturity of the partners and the management skills of the person who is allocated this role. These types of constraints are known as 'behavioural constraints' (Umble and Srikanth , 1990). Thus, the work habits, practices and attitudes of project personnel must be well known to this role to be able to successfully perform operational control.

Payments and balancing: on the operational level, the management of financial resources includes the responsibility for payments and invoicing, and the tracking of any anomalies in this regard, with feedback to the monitoring of resource usage. Feedback in normal cases can be provided as aggregate reports, however, in case of significant events, direct incident reporting is necessary. This role relies on information flow from the product side (VMP3) about incoming goods and services and maintains the day-to-day financial records of the project.

The management tasks listed above are likely to be allocated to more then one person (especially in larger projects), due to the diversity of skills necessary for this role.

12.7 Conclusion

This chapter has described the requirements specification of the management and control systems of enterprise entities. The management and control system may be modelled using a decisional model, which is a functional representation of management tasks. On the high level the model may be represented, as this chapter has demonstrated, using a decisional model employing GRAI Grids. On the detailed level, management tasks need to represent the decision tasks together with the information needs and resource needs, the controls (such as the details of decision frameworks and policies), and information produced (such as decision frameworks guiding other decision centres as well as internal or external information).

The detailed model of the decision system may be represented using several options, e.g. GRAI Nets, IDEF0 models, text, etc – as long as the necessary details are described, but the use of a formal language allows a more precise analysis of the specification. The requirement for using formal analysis to verify the correctness of the specification is especially necessary for the control system, which is algorithmic in nature. This part of the management and control system (called the *control* system) may be modelled using procedural languages (CIMOSA, IDEF3, Petri Nets, etc) and extended with matching state transition diagrams or state-charts.

The chapter has also discussed useful principles that may be followed in the design of the management and control system, such as the use of some form of agent model as a design pattern. Such principles improve the overall quality of the specification.

Finally, the chapter presented an example reference model for a network of partner companies created for the efficient joint delivery of projects. This reference model has given only an example to the method of describing a network of companies and their projects. Actual companies would have to develop their own decision system, depending on the agreed strategies and policies of the network in question, as well as the maturity level of the shared ICT infrastructure available or intended to be built.

References

Aerts, A., Szirbik, N. and Goossenaerts, J. (2000) Flexible Infrastructure for Virtual Enterprises, in Proceedings of the 4^{th} ICEIMT (November 2000).

Angell, I. and Smithson, S. (1991) Information Systems Management : Opportunities and Risks, London : Macmillan, 1991.

Baltrusch, R. (2000) Infrastructure Requirements for Virtual Manufacturing Enterprises. (Honours Thesis). Brisbane : Griffith University, School of Computing and Information Technology

Barbuceanu, M and Teigen,R. (1998) System Integration through Agent Coordination. In Handbook on Architectures of Information Systems. Berlin : Springer. pp 797-826

Bernus, P. (2001) Some thoughts on Enterprise Modelling, Int J. Production Planning and Control, London : Taylor & Francis, 12(2) pp110-118

Bernus,P., Nemes.L. (1999) Organisational Design: Dynamically Creating and Sustaining Integrated Virtual Enterprises, Proc IFAC World Congress, Han-Fu Chen, Dia-Zhan Cheng and Ji-Feng Zhang (Eds) Vol-A, London: Elsevier. pp189-194

Bernus, P. and Uppington, G. (1998) Co-ordination of management activities – mapping organisational structure to the decision structure, in Coordination Technology for Collaborative Application - Organizations, Processes, and Agents. in W.Conen and G.Neumann (Eds), LNCS 1364, Springer Verlag, Berlin (1998) pp25-38.

C4ISR Working Group (1998) Levels of Information Systems Interoperability. US Departmkent of Defense

Camarinha-Matos, L. and Afsarmanesh, H. (1998) Cooperative Systems Challenges in Virtual Enterprises, in Proceedings of the 2nd IMACS International Multiconference on Computational Engineering in Systems Applications (in CD), CESA'98, April 1998. p416

Cardoso, T., Camarinha-Matos, L. and Afsarmanesh H. (1999) Partners search and quality-related information exchange in a Virtual Enterprise. in Global Production Management IFIP International Conference on Advances in Production Management Systems, Boston : Kluwer pp76-84

Carnegie-Mellon Software Engineering Institute (2003) The Capability Maturity Model. Available: http://www.sei.cmu.edu/cmm/cmm.sum.html

Cooprider, J.G. and Victor,K.M (1993) The Contribution of shared knowledge to is group performance. ICIS 1993. pp285-299

Doumeingts, G.,Vallespir, B., Chen, D., (1998) GRAI Grid Decisional Modeling, in Handbook on Architectures of Information Systems. Berlin: Springer Verlag, pp313-338

Earl, M.J. (1989) Management Strategies for Information Technology. New York : Prentice Hall

EFQM (1992) EFQM Excellence Model. Bruxelles : European Foundation for Quality Management. http://www.efqm.org [note, the EFQM excellence model undergoes continuous improvement]

Emery, J. (1971) Organizational planning and control systems : theory and technology. 3rd edition, New York : Macmillan

FIPA (2002) FIPA Request Interaction Protocol Specification Document: SC00026H. Foundation for Intelligent Physical Agents.

Fox, M.S., Barbuceanu, M., and Teigen, R. (2000) Agent-Oriented Supply-Chain Management. International Journal of Flexible Manufacturing Systems. 12 (2/3) pp165-188.

Freidheim, C. (1998) The trillion-dollar enterprise : how the alliance revolution will transform global business, Reading, Mass.: Perseus Books

Frenzel, C. (1992) Management of information technology, Boston : Boyd & Fraser

Georgakopoulos,D., Schuster, H., Cichoki,A. and Baker,D. (1999) Managing Process and Service Fusion in Virtual Enterprises. Information Systems. 24(6) pp429-456

Globemen Consortium (Global Engineering and Manufacturing in Enterprise Networks) (1999) IMS project no. 99004. 1999, EU: IST-1999-60002.

Globeman21 GM21 D1.3 (1999) Final Report D1.3 on Models – Periodic Progress, Pedersen, J., Tolle, M. and Vesterager, J. (Eds.), Globeman 21 Consortium

Goranson, H.T. (1999) The Agile Virtual Enterprise, Wesport : Quorum Books

Handy, C. (1990) Trust and the Virtual Organization. in The Quest for Loyalty. Reicheld, F.(ed.), Harvard Business Review, pp 31-43.

Hardwick, M. and Bolton, R. (1997) The Industrial Virtual Enterprise. Communications of the ACM , 40(9) pp59-60

Harris, L., Coles, A., Dickinson, K. and McLoughlin, I. (1998) New Product Development Across Organisational Boundaries. in Jackson, P. (ed.). Virtual Working: Social and Organisational Dynamics. New York : Routledge. pp33-45.

Kiniry, J., Zimmerman, D. (1997) A Hands-On Look at Java Mobile Agents. IEEE Internet Computing, 1(4) pp21-30

Inmon, W.H. (2002) Building the Data Warehouse. (3rd edition) New York : John Wiley & Sons

ISA (2000) ISA–95.00.01–2000 – Enterprise-Control System Integration Part 1: Models and Terminology. Instrument Society of America. Research Triangle Park, North Carolina

ISO/IEC 12207.0 (1998a) Information Technology (IT)- Software Life Cycle Processes. International Standards Organisation

ISO/IEC (2002) ISO/IEC 15288: Systems life-cycle processes.

ISO/TC184/SC5/WG1. (2000). ISO/IS 15704: Industrial automation systems - Requirements for enterprise-reference architectures and methodologies.

ISO (2000) ISO 9000 Set of standards, ISO 9000 Collection 2: 2000 Edition.

STEP (ISO 10303) (1994) International Standard for the Exchange of Product Model Data. (10303 series of standards 1994-2002). Geneva : International Standards Organisation.

ISO/IEC TR 15504:1998 (1998b) - Software Process Assessment(SPICE)

Kalpic,B, Bernus,P. (2002) Business Process Modelling in industry - the powerful tool in enterprise management. Int J. Computer in Industry (3) pp299-318

Klenke, K. (1996) IS Leadership in Virtual Organizations. Proceedings of the 1996 Conference of ACM SIGCPR/SIGMIS. New York : ACM Press. pp22-35

Koestler,A. (1967) The Ghost in the machine. London : Arkana Books

Lorange, P. and Roos, J. (1992) Strategic alliances : formation, implementation, and evolution. Cambridge, MA : Blackwell

Mandelbrot, Benoit (1983) The Fractal Geometry of Nature. San Francisco: Freeman

McGrath, G.M. and More, E. (1999) Strategic Partnerships in the IT Sector: Trust, Power and Inter-Organizational Collaboration, Proceedings of the 10th Australian Conference on Information Systems, 1999. pp586-596

Nandhakumar, J. (1999) Consequences on Trust Relationships. in Jackson, P. (ed.), Virtual Working: Social and Organisational Dynamics. New York : Routledge. pp46-56.

Nemes, L. (2000) Foreword of the Proceedings of DIISM2000. Mo, J.P.T. and Nemes, L. (Eds) Boston : Kluwer

Pedersen, J.D. and van den Berg, R. (2000) Supporting Parner Selection for Virtual Enterprises. in Proceedings Proceedings of DIISM2000. Mo, J.P.T. and Nemes, L. (Eds) Boston : Kluwer.

Project Management Institute (1996) A guide to the project management body of knowledge. Upper Darby, PA.: Project Management Institute.

Rosettanet (2003),http://www.rosettanet.org

Shaw, R. (1997) Trust in the Balance, California : Josey-Bass Inc.

Spinosa,L. , Rabelo, R. and Klen , A. (1998) High Level Coordination of Business Processes in the Virtual Enterprise. In Proceedings of Prolamat'98 / The Tenth International IFIP TC5 WG-5.2 WG-5.3 Conference, September. pp725-736.

Umble, M. and Srikanth, M. (1990) Synchronous Manufacturing. USA : South Western Publishing Co. pp83-87.

Uppington, G. (1998) Identifying the road ahead: enhancing and extending the Enterprise Integration Identification Phase. (Doctoral Thesis, Brisbane : Griffith University).

Vesterager, J., Bernus, P., Larsen, L., Pedersen, J. and Tølle, M. (2000) Use of GERAM as basis for a virtual enterprise framework model, in Proceedings of DIISM2000. pp75-82

Vliestra, J. (1996) A Summary of the CIMOSA Reference Architecture. In Bernus, P, Nemes,L. and Williams,Y.J. (Eds.), Architectures for Enterprise Integration, London : Chapman & Hall. pp70-100.

Warnecke, H.J. (1993) The Fractal Company. Berlin: Springer.

van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P (1998) Reference Architecture for Holonic Manufacturing Systems: PROSA. Computers In Industry. 37(3) pp255-276

van der Aalst, W., van Hee,K., Schmidt, J.W (2002) Workflow Management: Models, Methods, and Systems (Cooperative Information Systems). Cambridge, MA. : MIT Press

Wiendahl, E., Engelbrecht, A., Hamacher, O. (2000) From Single Enterprises to Complementary Networks. in Proceedings of the DIISM2000 pp66-73

Zhou, M. (2000) The Architecture of an Internet-Based Virtual Community. In Proceedings of DIISM2000 pp61-65

13

RESOURCE REQUIREMENTS OF ENTERPRISE MANAGEMENT

Martin Zelm

CIMOSA Association

13.1 Introduction

Enterprises can only be well managed if their business processes are clearly defined, well understood and precisely controlled. All of these requirements can be achieved by employing business process based enterprise models, which describe both process functionalities and their dynamic behaviour. The description of functionality has to include inputs and outputs, the means of transformation - such as resources, and their constraints - as well as other relevant information. A prime goal of enterprise modelling is to analyse and control business operations and therefore to improve the deployment of resources (Vernadat , 1996).
In this Chapter it will be elaborated how resource requirements models (in the context of other enterprise models) can be used to support the mission of an enterprise and ensure its fulfilment. The standard ISO IS 15704 (ISO/TC 184/SC5 , 2000) defines functions of mission-fulfilment and functions of mission-control. Both function classes are involved in operating any enterprise - thus, both need suitable resources.

- One class comprises functions involved in fulfilling the mission, i.e. operating the processes that create the product or service. These would include all tasks of transformation, movement and storage of material, information and energy, as well as of goods in process, products and services. Enterprise architectures provide the capability to represent any process and its constituent activities involved in performing the established mission of the enterprise, in terms of providing the enterprise products and services to its customers.
- The other class comprises functions involved in managing and controlling the mission-fulfilment in order to achieve the desired economic outcome, or other gains that assure the viability or continued successful existence of the enterprise. These include the collection, storage and use of information to control the business processes (that is, to develop and apply nec-

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

essary changes to the business processes in order to achieve- and maintain their desired operation). Control includes all planning, scheduling, control, data management and related functions. (ISO/TC 184/SC5 (2000), section 0.2.3)

13.1.1 Scope of Resource Requirements Modelling

Capturing resource requirements means to define the required capabilities of resources that can fulfil the enterprise mission - also expressed in terms of objectives, goals, and business targets. Modelling the resource requirements allows enterprise management to better understand investments, cost, and resource utilisation within the organisation. Resource requirements will be derived from requirements of functional- and desired performance characteristics. In the context of enterprise architectures, resources are part of an integrated enterprise model. Resources are needed to execute business tasks or activities. With a model of resource requirements, management obtains transparency about the needed resources, capabilities, capacities and behaviour during the subsequent life cycle phases of preliminary design, detailed design and building.

Resources and their requirements can be distinguished depending on the time horizon[1] for which these resources are needed. For the long-term timeframe (or horizon), management defines the mission and the high-level strategic objectives and goals of the enterprise. To fulfil these objectives and goals, i.e., developing and providing products, strategic processes and resources (typically machinery, and personnel) have to be planned. For this purpose, a Master Plan must be developed, which determines the required resources, capacities and capabilities for the mission fulfilment.

For the short-term horizon, resource requirements are continuously determined as part of the operational control of the enterprise, namely by its resource management and control, trying to balance and optimise the available resources in the supply chain. Ideally, this process is accomplished as part of the usual practices of enterprise management; thus, resource deployment at any one time is consistent with the master plan. Should enterprise management encounter frequent events where these usual practices cannot be satisfied by current resources or resource management practices, the enterprise needs to take strategic action (part of which may be a change to the master plan). Signs of such circumstances may be, for example, the inability to use resources to the full extent, the inability to meet deadlines, requests from partners in the supply chain, or lost business.

Typical resource management activities are: flexible resource adaptation (production planning and scheduling), resource (re)configuration, negotiation with partners, suppliers or agents, and in general managing the supply chain in the

[1] the term 'horizon' used hereafter interchangeably with 'timeframe'

changing business environment. Balancing of required and provided capabilities will lead to optimised resource identification and usage.
A special remark on resource reconfiguration needs to be made here. As long as the cost and speed of reconfiguration is within the acceptable limits of the company's day-to-day operational objectives, resource reconfiguration may be regarded as part of the operational activities of the enterprise. However, if the company experiences strains as previously exemplified, this can be a significant event - which the enterprise needs to consider as a requirement to modify its master plan (perhaps by making significant changes to its resources (with greater flexibility and less cost and time for reconfiguration) or, as sometimes is the case, improving on the reconfiguration process that is part of resource management). As part of the master planning process, flexibility to reconfigure is regarded as a designed property.
As a third alternative, resources can be acquired on a temporary basis, through the formation of a virtual enterprise built on a project-type organisation and relying on a network of enterprises, where resources are used out of a larger pool of resources. Clearly, the decisions about virtual enterprise projects can only be made if one is able to correctly analyse the resource requirements.
If the analysis of the resource requirements shows that the necessary life cycle activities can only be carried out at a slow rate, then management needs to decide whether to pursue this task at all. Depending on the time horizon of the business scenario (whether a short-term opportunity, or a long-term investment) the resource requirements will be different. Below it is described how to determine resource requirements and to prepare the enterprise to have the capability to do this analysis quickly. This capability is used in both short- and long term resource development.

13.1.2 Resource Models Within a Reference Architecture

Resource models play an important role in enterprise architectures. Enterprise modelling frameworks, as those of GERA[2], CIMOSA[3], GRAI[4], Zachman provide guidance as to the necessary models in enterprise engineering. Several sets of enterprise modelling languages (such as those of CIMOSA, and the IDEF family of languages) contain suitable constructs for carrying out this modelling task. Using these languages, the modelled objects may then be represented from different viewpoints, and at different levels of granularity, as the corresponding life cycle phases require (IFAC-IFIP Task Force , 1999; Cimosa Association , 1996; Doumeingts et al. , 1998) (Menzel and Mayer , 1998).

[2] Generalised Enterprise Reference Architecture, described as part of GERAM (Generalised Enterprise Reference Architecture and Methodology), which is presented in Annex A of ISO/IS 15704
[3] Computer Integrated Manufacturing - Open System Architecture, http://www.cimosa.de
[4] Graphs with Results and Activities Interrelated

In GERA (and other compliant modelling frameworks), resources models are visible in the Resource view (refer Chapter 2, Fig. 2.10). These resources are then employed to execute activities and processes as described in corresponding other modelling views. Hence, the resource model is part of one integrated enterprise model comprising all views.

13.1.3 Resource Types

The purpose of resources is to perform tasks or activities in a process that in turn fulfils some enterprise mission. Various activity types need to be performed: Humans carry out business tasks or provide services to customers, machines produce products, computers execute programs as well as store and retrieve data. Resources may be classified into:

- human resources (with profiles including capabilities, skills, knowledge),
- machine technology resources (manufacturing equipment, NC machines, conveyors and storing facilities, as well as computing and communication devices),
- software application resources (such as data bases, application programs, controllers, tools, agents , and knowledge captured in them),
- financial resources, energy or time (Vernadat , 1996).

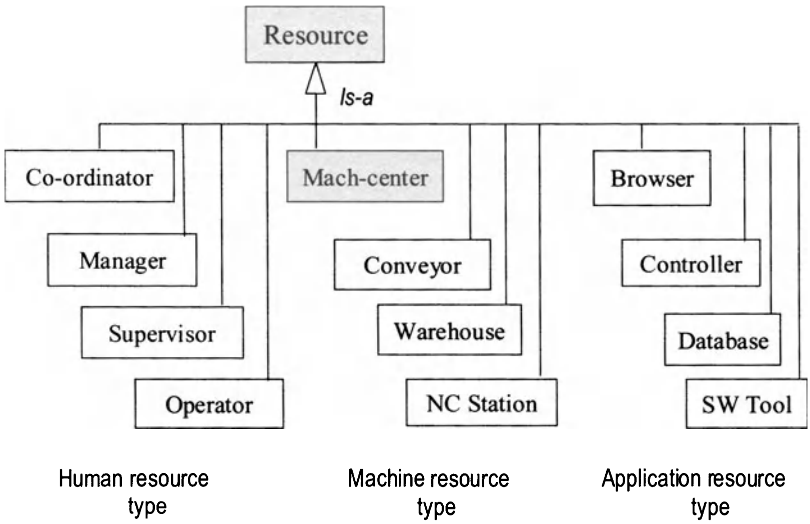


Fig. 13.1. Some Resource types

Since this Chapter discusses the use of resources regarding their ability to perform enterprise activities, only the first three resource categories will be

considered in more detail. Figure 13.1 illustrates a possible classification of resources.

13.2 Languages to Model Processes and Resources

The most commonly used enterprise modelling languages have been reviewed and compared in terms of their modelling principles, coverage and capabilities (Vernadat , 1997). Out of the list of enterprise modelling languages, only two provide explicit constructs to model enterprise resources and their requirements in detail, namely Integrated Enterprise Modelling (IEM - Spur et al. (1996)) and CIMOSA (and its implementations in enterprise engineering tools[5]). Table 13.1 shows some selected modelling capabilities relevant for resource modelling.

Table 13.1. Modelling languages IEM and CIMOSA

Modelling Language	Generic Model	Model views available	Life Cycle Support	Resource modelling constructs	Process and Resource separated in modelling language	Supported Standards
IEM	Yes	Yes	Limited	Resource	Yes	ENV 12204
CIMOSA	Yes	Yes	Yes	Capab. Set, Resource	Yes	ENV 12204

Both modelling languages comprise concepts to represent objects from different viewpoints, at different levels of genericitity and during different life cycle phases. Both have explicit constructs to model resources, and provide separation between processes and resources, enabling independent modelling of either one or the other. Both languages are supported by commercial modelling tools[6].

[5] Some implementations, e.g. in the FirstStep tool, included for convenience an explicit (and extendable) classification of resources - similar to the classification presented in Fig. 13.1. The CIMOSA resource modelling language itself includes the basic resource concepts (Active and Passive resource) - these can be further classified according to the needs of the modelling task.

[6] e.g. CIMOSA by FirstStep (Levi and Klapsis , 1999) and IEM by MO^2GO (Mertins and Jochem , 1998)

Standardisation plays an important role in selecting resource modelling languages. CEN TC 310[7] has developed the European Standard prENV 12204 *Systems Architecture - Constructs for Enterprise Modelling* with a set

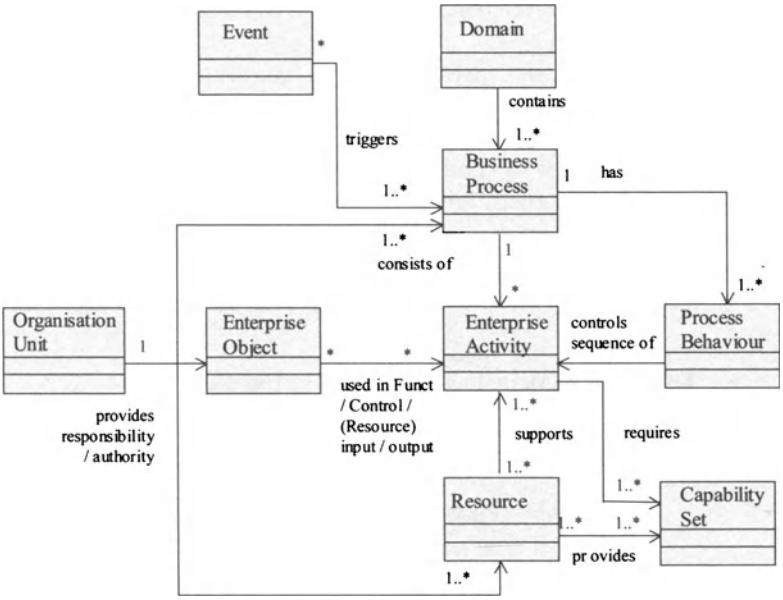


Fig. 13.2. Modelling language constructs proposed for ENV 12204

of user oriented modelling language constructs providing definitions and templates for modelling resources and capability sets and their relationships (CEN TC310 WG1 , 1995). This set of modelling constructs is shown in Fig. 13.2.

13.2.1 Resource Modelling with IEM

IEM uses an object-oriented language for the description of information and function views. These views are defined on a single object model of an enterprise system. The core of the model comprises two views: the Business Process model and the Information model. Production and other related activities are modelled as functions and business processes that are related to objects. The execution of functions or business processes results in a change of attribute values of the respective objects (Spur et al. , 1996).
Each enterprise model is built from the generic object classes, *Resource, Product* and *Order* from which sub-classes are derived. Necessary enterprise data

[7] Comité Européen de Normalisation (European Committee for Standardization), Technical Committee 310

and business processes are assigned to these sub-classes. Relations between these objects are also represented. This results in a comprehensive registration of all tasks, including process organisation, enterprise data as well as production facilities and information systems on any desired level of detail.

13.2.2 Resource Modelling with CIMOSA

The modelling framework of CIMOSA (Cimosa Association , 1996) defines a cube with three levels of genericity (generic, partial, particular); these modelling levels follow the GERA life cycles and are called requirements, design, and implementation. These correspond (refer Chapter 3) to the GERA Requirements, Preliminary design and Detailed design phases (CIMOSA also defines the release to operation phase, not represented in the CIMOSA cube, which corresponds to the GERA Build phase). CIMOSA has four model views: function, information, resource, and organisation. The concept of views allows the modeller to work with a subset of the facts represented in the enterprise model, rather than working with the complete model.
CIMOSA distinguishes two categories of enterprise resources:

- Active resources - called Functional Entities that have some degree of intelligence or autonomy. Functional Entities have the capability to send, receive, store and process information. The main characteristic property of Functional Entities is that they can be controlled (i.e., they are able to receive control information).
- Passive resources, also called Resource Components, have no intelligence or autonomy. They are always employed by-, or incorporated as components into an active resource. Examples of active resources are software applications, computer-controlled machines, robots etc. Examples of passive resources are tools, operator-driven machines, trucks, etc.

13.2.3 Resource Requirements Modelling

The Pre-Standard ENV 12204 provides one single construct to model resource requirements, namely the capability set, which defines a set of capabilities required by Enterprise Activities. Capabilities are needed for resources to be able to carry out one or several enterprise activities. Figure 13.3 shows the template of the modelling construct Capability Set (CEN TC310 WG1 , 1995).
Capability Sets are defined for Human, Machine and Application Capabilities (Cimosa Association , 1996). For example, Human Capabilities, which are operation related, embrace a set of human competencies, e.g. the ability *and* the qualification to perform a task - instead of the capability of just having the qualification to perform a task. Additional capabilities might have to be added to this set, such as *cognitive* abilities, e.g., the ability to plan, to decide, to solve problems, etc.

Header Part
IdentifierName and/or number
Class NameCapability Set
(Types Human, Machine or Application Capability Set)
Sub-class NIL
Functional descriptionText
Design AuthorityOrganisational unit
Identifier
Body Part
Function related NIL or list of (Capability)
Object related NIL or list of (Capability)
Performance related NIL or list of (Capability)
Operation related NIL or list of (Capability)
Pre-defined or user defined attributes:
List of (enterprise object attributes and or Enterprise object identifiers)

Fig. 13.3. Template of Capability Set (ENV 12204)

The Capability Set construct can be used to define aggregated or detailed Capabilities, according to the level of detail defined for a particular Enterprise Activity. Table 13.2 shows categories of capabilities. Usually, a set or a profile of capabilities is required. Note that, in addition to having all individual capabilities to perform a set of activities in a process, an additional one is necessary, namely the capability to combine available capabilities / skills in a suitable manner to perform a process (or an activity) that requires the given capabilities (according to Chapter 4), which explains the relationship between individual capabilities and core competencies, for example.

Table 13.2. Capability categories

Capability	**Criteria**	**Examples**
Function related	Having the functionality to perform one or more Enterprise Activities	Business transformation, metal cutting, AGV[8] transport
Object related	Ability to process the input(s) of an Enterprise Activity	Min/max value range of inputs
Performance related	Performance required by an Enterprise Activity	ROI[9] within 5 years, % of equipment availability
Operation related	Constrains on operation related attributes imposed by an Enterprise Activity	Education level, personnel skill, working hours per month

Resource requirements modelling is intrinsically connected to function- and process modelling. Thus, we suppose that in parallel with resource requirements modelling, a function model (or right away a process model) is being created. When such a model is developed on the preliminary design level (e.g. using CIMOSA modelling constructs or IDEF0 constructs), the resources (or mechanisms) to carry out the activities are identified. While on the preliminary design level resources are only named (but not selected), on the detailed design level decisions about the actual resources must be made.
Thus, while doing function or process modelling, the emphasis is on determining the capabilities that the identified activities / processes will *need.* This leaves plenty of design decisions to be made in the detailed design phase. An Operation Analysis may be performed in order to determine the required capabilities. The outcome of operation analysis is twofold:

- on one hand the identification of the required activities (captured in the function or process model), and
- on the other hand these activities must be characterised as for their capacity, quality, cost, throughput, flexibility, and the set of functions, which, if suitably combined, can perform the activity (see Table 13.2).

It is to be noted that the level of detail of function or process models is limited to levels where the activity / process can be performed by only one active resource.
Operation Analysis is concerned with the detailed description of the activities defined in the previous modelling step, where information and material inputs have been identified, but without detailed characterisation of their qualities or quantities. An enterprise activity is considered as the point of use of information and material, of required capabilities and the point of creation of new information and material in the enterprise operation. This allows to identify all sources and sinks of operational information as well as material. Necessary capabilities are therefore dependent on the information and material inputs and outputs of these activities.
Operation Analysis may be followed by an Information and Material Flow Analysis, in which the relevant object views - inputs and outputs of enterprise activities - are described with their attributes, consolidated, and arranged into hierarchical object structures. The identified enterprise objects and their relations are represented in an Entity Relationship schema or a class diagram. To consolidate this model, the consistency of the model may be checked via a structured walkthrough, or model animation. While model animation is not possible on the basis of function models, a subsequently developed process model is already executable and therefore simulation tools can be used.
Such executable models may be analysed in several manners. Simulation allows multiple test cases to be executed and statistical information gathered,

[8] Automated Guided Vehicle
[9] Return on Investment

in order to determine whether the process has the desired characteristics. Also, exccutable models may be analysed from the point of view of deadlocks or other undesirable structural characteristics, such as for the identification of hot-spots in the process where a single point of failure may cause undesired disruption. Also sensitivity analysis can be carried out, either through simulation or preferably through analytic means, in order to determine the sensitivity of output characteristics relative to statistical changes in the quantitative properties of inputs and of resources. Such information may then be used for risk analysis.

As an example, let us assume that the desired nominal performance of a resource is expected to change 10% upward or downward. Sensitivity analysis can identify how much such a change in input volumes or resource performance will effect the production.

The problem of analytic risk analysis is that while changes in a single quantity may be determined on the basis that all other inputs and resource capacities are on their nominal level, this is often not possible if *multiple* variables change simultaneously.

In the case of fears that simulation may not have provided enough information to determine that only acceptable risks are taken, either the structure of the process must be changed (so as to allow conclusive evidence to be drawn on the risk levels contended), or resource characteristics need to be upgraded to deal with unexpected combination of lows and highs in multiple process variables (such as input volumes and resource performance).

The same models can be used for the analysis of other disruptions, either due to malfunction, or scheduled maintenance.

13.3 Resource Requirements Determination in the Virtual Enterprise

One way to increase the flexibility of resources is to develop the capability to participate in virtual enterprises. While a detailed account of virtual enterprises is given in Chapter 7, the following discusses the resource requirements analysis aspects of such enterprises. As virtual enterprises are formed through linking the mission fulfilment and management processes or participating individual companies, it is a natural extension to resource modelling to base resource requirements modelling on activity and process models of virtual enterprises. These models of a virtual enterprise allow the simulation and analysis of the processes carried out by the virtual enterprise and evaluate their benefits and shortcomings (Kosanke et al. , 2000). One prerequisite however is that the requirements related to the virtual enterprise's mission should be well established.

Figure 13.4 shows the process model of a virtual enterprise. Company **A** to Company **D** have established for a limited period to co-operate for achieving a common goal (Weston et al., 1997). The process stream is established by a

set of processing nodes or a network of activities. These nodes still reside in their company environments - within their own organisation - but are linked into business processes cutting across several organisations.

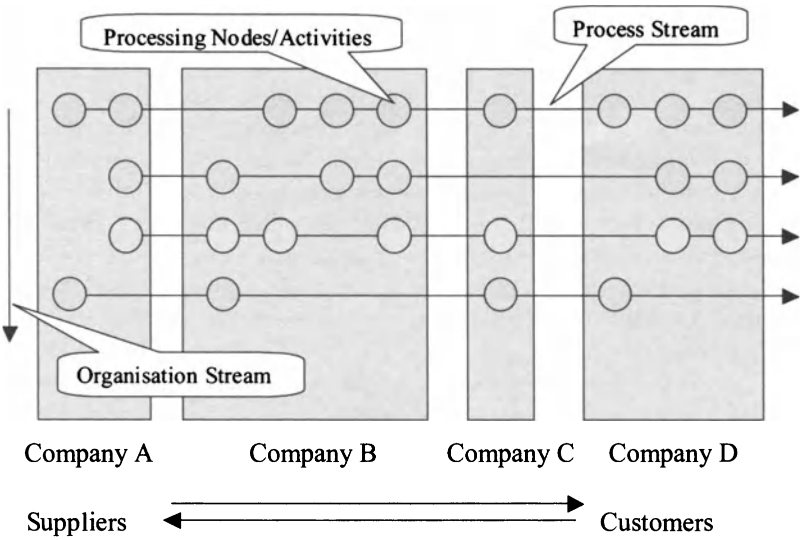


Fig. 13.4. Inter- and intra-enterprise operation

The resource requirements can be determined at each process node in terms of their required capabilities. For each activity, the required capabilities can be identified and described. The accumulated sum of all capabilities makes the total resource requirement of the virtual enterprise. Using this requirements model, the availability of resources can be analysed - in each company, for each process or in the entire virtual enterprise. For example, by analysing the occurrences of activities an estimate of resource balancing, trade-offs, bottlenecks can be obtained at the requirements level. This model will mainly serve as an input for the design model.
The application of enterprise models is limited by the time constraints of the lifetime of the virtual enterprise itself. The concept of resource requirements collection using enterprise models is only meaningful if the creation of the complete model can be done on short notice, because the model creation time is very short compared with the duration of the virtual enterprise. This requirement can only be met if the partners in the virtual enterprise have models of their process contributions readily available and if these models are compatible and allow for model inter-operability.
A particular problem of function and process models in the context of virtual enterprises is the availability of such models for each participating company. In order to overcome this problem, companies wishing to be ready for virtual

enterprise formation need to have models in the same formats, and be prepared to share these models for the purposes of analysis and simulation. Given that a detailed model of the company's operation is part of its competitive advantage, companies are usually *not* prepared to share them. The formation of enterprise networks, through pre-agreements and limitations to the disclosure of information is a prerequisite to the use of activity and process models, especially if they are enriched with resource models.

If function and process models are not available to the level of detail that allows simulation or overall analysis of the supply chain, then distributed simulation techniques may have to be used. In distributed simulation each company runs their simulation on their own detailed model, while these simulators exchange limited information through their interfaces. Even in this case pre-agreements are needed, since it is known from control theory (systems identification) that a sufficient amount of simulated cases allows the behavioural contents of a black box to be reconstructed (called *identification* in control theory).

Another problem to be attended to is that the participants of the eventual virtual enterprise are not necessarily fixed at the time of design. Therefore, potentially a great number of variations of process models may be expected. One way out of this dilemma is for the creator of the virtual enterprise to establish a complete process model on the basis of key participants (those who are fixed) and design a process on the basis of industry experience, selecting the rest of the participants based on their ability to fit into this stipulated process.

The problem is similar to process planning in manufacturing, where a number of feasible ways exist to manufacture a particular product. Given the product mix and the available capacities, the process selected which fits best to this already constrained problem. While overall optimum cannot be achieved, a reasonably good process can be selected from a few variants in this way.

At the level of requirements, the analytic and simulation models are not yet used to allocate resources to activities. This will be the task of subsequent life cycle phases, but resource requirements models can shed light on the possible constraints that process characteristics may have to comply with. Often, such analysis revels weaknesses in the process model (e.g. through sensitivity analysis), which must be rectified through process improvement. The preliminary and detailed design activities will then have to work in this constrained design space.

13.4 Outcomes of Resource Requirements Determination

Enterprise modelling will play a significant role in the identification of the relevant information as well as provide decision to support itself. Depending on the specified mission, the requirements of human, software and hardware resources can be modelled and thereby determined. This will allow the designer

to investigate and evaluate process alternatives and allow the establishment of business performance indicators, control figures, model based business cases, as well as libraries of partial re-usable models. Modelling supports the new organisation types of virtual or extended enterprise. It leads to transparency of the business scenario, less business risk and to better overall performance. Modelling resources is one aspect of a requirements model. It presents an initial step to capture and manage resources during the requirements life cycle phase. A rich comprehensive requirements model is the prerequisite for an effective design model and finds its application in subsequent life cycle phases. The subsequent preliminary design, where the resource architecture is determined and detailed design where actual resources are allocated to activities, will rely on the exact statement of functional, information, capability and capacity requirements determined in the Requirements definition life cycle phase.

References

CEN TC310 WG1 (1995) ENV 12204: Advanced Manufacturing Technology. Systems Architecture. Constructs for Enterprise Modelling. Brussels: Comité Européen de Normalisation

CIMOSA Association (1996) CIMOSA: Open Systems Architecture for CIM, Technical Baseline - Part I, private publication.

Doumeingts, G., B. Vallespir, D. Chen (1998) GRAI Grid Decisional Modelling. In Handbook of Information Systems, Bernus, P., Mertins, K. and Schmidt, G. (Eds.), Springer-Verlag, Berlin. pp. 313-337.

Kosanke,K., Vernadat, F.B. and Zelm, M.(2000) Enterprise Engineering and Integration in the Global Environment, Camarinha-Matos, L.M. et al (Eds.) Advances in Networked Enterprises – Virtual Organisations, Balanced Automation, and Systems Integration (selected papers from IEEE/IFIP BASYS 2000), Kluwer, Berlin, 2000-09-27/29.

IFAC-IFIP Task Force (1999) GERAM: Generalized Enterprise Reference Architecture and Methodology, Version 1.6.3, IFAC-IFIP Task Force on Architecture for Enterprise Integration.

ISO/TC 184/SC 5, ISO IS 15704 (2000) Industrial automation systems - Requirements for enterprise reference architectures and methodologies; Secretariat: ANSI/NEMA, Géneve 2000

Levi, M., Klapsis,M.P. (1999) FirstSTEP process modeller - a CIMOSA-compliant modeling tool. Computers in Industry. 40 (2-3). pp 267-277

Menzel, C. and Mayer, R. (1998) The IDEF family of languages. In Handbook of Information Systems, Bernus, P., Mertins, K. and Schmidt, G. (Eds.), Springer-Verlag, Berlin. pp. 209-241.

Spur, G., Mertins, K., Jochem, R. (1996) Integrated Enterprise Modelling. Berlin : Beuth-Verlag.

Mertins, K., Jochem, R. (1998) MO^2GO. In Handbook of Information Systems, Bernus, P., Mertins, K. and Schmidt, G. (Eds.), Springer-Verlag, Berlin. pp. 589-600.

Vernadat, F.B. (1996) Enterprise Modelling and Integration, Chapman and Hall, London

Vernadat, F.B. (1997) Enterprise Modelling Languages. in Proc ICEIMT'97. Kosanke, K. and Nell, J.G. (Eds), Berlin : Springer. pp 212-224.

Weston, R.H., de la Hostria, E., Kosanke, K. and Noxon, E.R. (1997) Business Benefits from Enterprise Integration, in (Eds) Enterprise Engineering and Integration: Building International Consensus, Proceedings of the ICEIMT'97, pp 152-162, Springer.

14

ENTERPRISE MODELLING

Michael Grunninger

NIST**

14.1 Applications of Enterprise Modelling

An enterprise model is a computational representation of the structure, activities, processes, information, people, behaviour, goals and constraints of a business, government or other enterprise. An enterprise model can be both descriptive and definitional and it may cover both 'what is' and 'what should be'. The role of an enterprise model is to achieve model-driven enterprise design, analysis, control[3] and evaluation. We will begin by considering these applications of enterprise modelling and the requirements that they impose on any formal approach to the representation and specification of enterprise models.

14.1.1 Enterprise Integration

Interoperability among enterprise applications is often hindered because the applications use different terminology and representations of the domain (Ciociou et al. , 2001). These problems arise most acutely for systems that must manage the heterogeneity inherent in various domains and integrate models of different domains into coherent frameworks (Fig. 14.1).

For example, consider concurrent engineering. A design engineer creates a product specification using a CAD system; this design must be integrated with the company's product data management (PDM) system, which will represent not only the product's features and geometry, but may also include notions such as design intent, additional product requirements and a bill-of-materials (BOM) decomposition. This design must be shared by the engineer with the process design team (which uses the BOM to specify a set of manufacturing processes whose final output may be a product with the desired features). Any version of the process design may be shared with the process

** National Institute of Standards and Technology, US

[3] using executable models (e.g. such as the ones CIMOSA aims to produce)

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

planning team, which specifies the various machines, tools, and materials that will be required by the manufacturing processes. If the process design team identifies any problems with these processes, they must be communicated to the product designer, who may need to modify the design to guarantee manufacturability. The production planning team will need to share the process plan, since it must be included within the production plan, together with the process plans of other products. Schedulers take the production plan and add further constraints on the occurrence of various processes. If either the production planner or scheduler discovers a problem (such as unanticipated bottleneck resources), the underlying process plan or production plan may need to be revised upstream (by earlier teams).
Such integration occurs, for example, in business process re-engineering, where enterprise models integrate processes, organizations, goals and customers. Even when the applications involved use the same terminology, they often associate different semantics (meaning) with the terms. This clash over the meaning of the terms prevents the seamless exchange of information among application programs. Typically, point-to-point translation programs are written to enable pair-wise communication between specific applications. However, as the number of applications used in enterprises has increased and the information has become more complex, it has been more difficult for software developers to provide translators between every pair of applications that must cooperate. What is needed is some way of explicitly specifying the terminology of the applications in an unambiguous fashion, so that every application could refer to such common meanings – whether directly or indirectly.

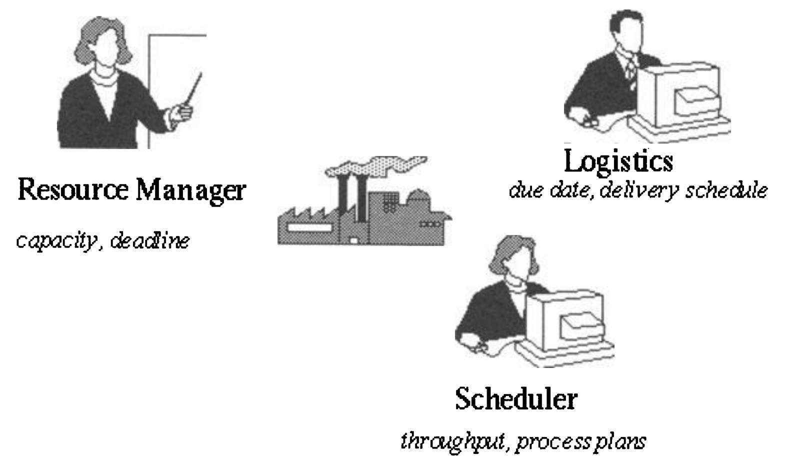


Fig. 14.1. The challenge of interoperability

14.1.2 Reusability

Knowledge bases that capture the domain knowledge of engineering applications are often tailored to specific tasks and projects. When the application is deployed in a different domain, it does not perform as expected, often because assumptions are implicitly made about the concepts in the tailored application, and these assumptions are not generic across domains. For example, machine models are often designed for a particular set of properties specific to particular (types of) machines, rather than characterizing generic properties of machines, such as concurrency constraints, setup activities, and operating conditions. Reusability can be achieved through shared understanding of generic concepts that span across multiple projects, tasks and environments[4].
One of the bottlenecks in enterprise engineering is enterprise model *acquisition*. Models are often constructed for single projects, with little reuse. The dream is to create new models from a repository of existing (partial or particular) enterprise models. Partial models would then be combined into an integrated model of the entire enterprise, thus supporting the iterative refinement/elaboration of the enterprise model. Existing models could be modified to capture the challenges posed by new situations, while templates for various classes of enterprises would allow rapid modelling through instantiation.

14.1.3 Enterprise Analysis

An integrated enterprise model provides the language used to specify an explicit definition of an enterprise. The easiest application of such an enterprise model is in checking the consistency of the enterprise model with respect to additional constraints. This may be an *internal* consistency check (in which the enterprise model itself is tested to identify enterprise design problems) or it may be an *external* consistency check (which compares the intended behaviour of the enterprise as expressed in the model with the actual behaviour of the enterprise).
For re-engineering, enterprise engineers need to explore alternative models in the design of enterprises, spanning organisational structure and behaviour. In order to reason about alternative designs for enterprises, they need to reason about different possible sets of constraints for enterprises within the language. Such reasoning can often be expressed by queries whose answers can be deduced from the enterprise model – e.g.: Can a process be performed in a different way, or can the enterprise achieve some goal in a different way? Can the constraints in the enterprise be relaxed, such that we can improve performance or achieve new goals?

[4] **NB** Models with various degrees of reusability may also be achieved by using *a mix* of generic and particular properties. Such models are considered *partial* - i.e., where only some properties are particularized, the rest being generic

Enterprise engineers also need to be able to determine the impact of changes on all parts of the enterprise, which involves *hypothetical* reasoning. For example, if one of the policies is relaxed, how will this affect the quality of products or services provided by the enterprise? If a new kind of machine is purchased, how will this affect the activities that are performed? If the enterprise changes the activities that are performed, how will this change resource consumption? A related problem is the use of benchmarking in the re-engineering process. In benchmarking, performance is compared between enterprises and then processes and practices are adopted from enterprises that are the best performers. However, not all practices can be adopted from other enterprises; the key is to realize that one must identify opportunities for improvement by analysing the successes and failures of similar enterprises. Herein lies the problem – what is a *similar* enterprise? What is compared among enterprises when using the benchmarking approach? Goals and activities cannot be compared among enterprises unless all constraints and assumptions about the enterprise and its environment are made explicit.

14.1.4 Model-driven Enterprise Operation

In many Business-to-Business (B2B) E-commerce applications, enterprises face the same integration problems among enterprise systems, although it often involves the semantic integration of intelligent agents that are intended to automate business processes. Emergence of the 'Semantic Web' technologies carries the promise of new advances in the area of inter-enterprise and B2B interoperability, as it has a potential to provide a shared basis for capturing semantics of enterprise-level activities and concepts. However, there is currently a gap between the business processes and organizational structures that are defined within an enterprise model and the enterprise applications that either automate or support these business processes. The behaviour of agents within the enterprise often does not conform to the constraints defined within the enterprise model.

14.2 Ontologies

To support these applications of enterprise models, there has been an increasing interest in Generic Enterprise Models (GEM) (Gruninger and Fox , 1998). A GEM is an object library that defines the classes of objects that are generic across a type of enterprise, such as manufacturing or banking, and can be employed (through instantiation) in defining a specific enterprise. The benefit of such an enterprise model is that the object library supports reusability and integration through a common conceptualisation. The problem is that different representations of the same information may be based on different assumptions about the world, and use differing concepts and terminology – and conversely, the same terms may be used in different contexts to mean

different things. Often, the loosely defined natural-language definitions associated with the terms will be too ambiguous to make the differences evident, or will not provide enough information to resolve the differences.

To address these challenges, various groups within industry, academia, and government have been developing sharable and reusable models known as *ontologies.* The most commonly quoted definition of an ontology is "*a formal, explicit specification of a shared conceptualisation*" (Gruber , 1993). In this context, a *conceptualisation* refers to an abstract model of how people think about things in the world, usually restricted to a particular subject area. An *explicit specification* means that the concepts and relationships of the abstract model are given explicit names and definitions. The name is a term, while the definition is a specification of what the term means. *Formal* means that the specification is encoded in a language whose formal properties are well understood – in practice, this usually means logic-based languages. Formality is an important way to remove ambiguity that is prevalent in natural language; it also opens the door for automated processing of semantics.

It is common to refer to taxonomies, thesauri, data dictionaries, data models and other representations as ontologies, despite their lack of formality. Nevertheless, there is a core essence that is common to virtually all uses of the term 'ontology'. This core is that there are two essential components of any ontology:

- a vocabulary of terms, and
- some specification of meaning for the terms.

What distinguishes the many types of things that people refer to as ontologies is the degree and manner of specifying meaning. This gives rise to a kind of continuum of *kinds of* ontology (Gruninger and Uschold , 2002). At one extreme, we have very lightweight ontologies that may consist of terms only, with little or no specification of meaning (the degenerate case of an ontology). At the other end of the spectrum, we have rigorously formalized logical theories, which comprise the ontologies (refer Fig. 14.2). Moving to the right along the continuum, the amount of meaning specified increases (thus reducing ambiguity), the degree of formality increases, and there is increasing support for automated reasoning.

In the simplest case, the semantics are *implicit* only. Meaning is conveyed based on a shared understanding derived from human consensus. A common example of this case is the typical use of XML tags, such as *price*, *address*, or *delivery date.* Nowhere in the XML document, nor anywhere else, does it say what these tags mean (Cover , 1998). However, if there is an implicit shared consensus about what the terms mean, then people can embed this implicit semantics in screen-scrapers and wrappers. Online travel agents and booksellers routinely do this to find the best deals. From the perspective of mature commercial applications on the Web, this is the current state of the art. The disadvantage of implicit semantics is that they are rife with ambiguity because people often *do* disagree about the meaning of a term.

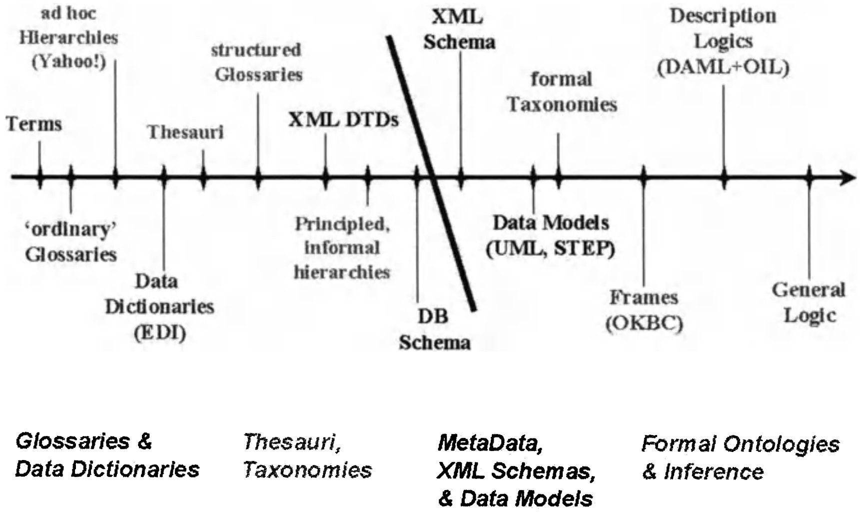


Fig. 14.2. Kinds of Ontologies – There are many kinds of things that people call ontologies. Moving to the right there is reduced ambiguity and increased amount of meaning, formality and support for automated reasoning. Not everyone agrees on what is, or what is not an ontology. A line has been arbitrarily drawn, to the right of which there is quite broad agreement that they are ontologies. To the left of the line, the agreement on ontologies is more controversial.

At the next point on the continuum, the semantics are explicit and are expressed in an *informal* manner, often as text in a specification document. Until the natural language processing problem is solved, only humans can make direct use of informally expressed semantics. Examples of informal semantics expressed in text specification documents: 1) the meaning of tags in HTML, for example, <h2>, which means second level header; 2) the meaning of the subClassOf relationship in the RDF Schema language; and 3) the meaning of expressions in programming languages, such as Java.
The main disadvantage of implicit semantics is that there is still much scope for ambiguity. This decreases one's confidence that two different implementations (say of RDF Schema or Java) will be consistent and compatible. Each Java implementation may be different in subtle ways. Users may notice 'features' and start depending on them. This can result in problems if interoperability is required, or if implementations change.
Finally, there is the possibility of explicit, formally specified semantics that are intended for automated inference. Formally specified ontologies use a logical language, such as DAML+OIL (Hendler and McGuinness , 2001) and the Knowledge Interchange Format (KIF) (Genesereth and Fikes (1992) and Hayes and Menzel (2001)). A formal ontology consists of a set of sentences in

this underlying logical language. Within mathematical logic these sentences are also known as *axioms,* so that a formal ontology is also said to be *axiomatised.* The idea is that when new terms are encountered, it is possible to automatically interpret something about their meaning and thus how to use them. In order for an enterprise model to support such inference, it must provide a set of rules of deduction together with a set of axioms. For example, given an axiom stating that the *works-for* relation is transitive:

If X works-for Y and Y works-for Z, then X works-for Z

and the facts that Alice *works-for* Bob and *Bob works-for* Carol, an inference system would be able to automatically deduce that Alice *works-for* Carol.

14.3 Desiderata for Enterprise Modelling Ontologies

Based on the scenarios in the first section, what are the requirements that must be satisfied by ontologies for enterprise modelling both in terms of their formality and in terms of their content?

14.3.1 Verification and Semantic Integration

If we are to support scenarios in which enterprise applications *automatically* share and reuse information, we must provide guarantees that the applications completely understand each other within the context of their shared domains. For example, reusability means that enterprise models for new or re-engineered enterprises can be specified by extending (from generic- and/or partial models) or modifying (from partial- and/or particular models) concepts defined in an existing ontology. Given an enterprise, the modelling task consists of using the ontology to specify a set of expressions that captures the behaviour of the enterprise. How does the engineer determine that he or she is using the correct set of predefined concepts for modelling the enterprise? The engineer must understand the meaning of the ontology that is being reused in the way that was intended by the designer of the ontology in question.

The key notion in both semantic integration and reusability is that one must somehow guarantee that the intended meaning of the terms in an ontology is the only meaning for those terms that is consistent with the content of the ontology in conjunction with the semantics of the ontology representation language. Even in cases where the ontology is not explicit, one can examine the behaviour of the application and consider it to be completely dependent on the inferences it makes, which inferences in turn are completely dependent on the axioms in its ontologies and other internal knowledge. In effect, one may consider an application's behaviour to be constrained by the semantics of its ontology, because any inferences made by the application must conform to those semantics (refer Fig. 14.3. If an application does not behave as expected,

or does not retrieve relevant information, or does not make correct inferences, then it may be concluded that the application in question does not share its semantics with other applications. In this way, one can observe the application's behaviour and infer information about the semantics the application is using.

One must therefore explore the ramifications of explicit formally specified semantics and the requirements that this approach imposes on both the ontology designer as well as the enterprise application designer. There is a need to characterize the relationship between the intended meaning of an application's terminology and the possible meaning of terms in the application's ontology that must hold to support complete semantic integration and reusability.

In most current ontology research, the languages for formal ontologies are closely related to mathematical logic[5], in which the semantics are based on the notion of an *interpretation*. If a sentence is true in the interpretation, we say that the sentence is *satisfied* by the interpretation. If every axiom in the ontology is satisfied by the interpretation, then the interpretation is called a *model* of the ontology. With a formal ontology, the application's knowledge is specified as a theory, so that a sentence is *consistent* with that theory if there exists a model of the theory that satisfies the sentence; a sentence can be deduced if it is satisfied by all models of the theory. Therefore, the application's behaviour (and hence the semantics of the application's terminology) can be characterized by this implicit set of models, which will hereafter be called the set of *intended models*.

This characterization may be used to evaluate the adequacy of the application's ontology with respect to these intended models. An ontology is *verified* if and only if the set of models of the axioms of the ontology is equal to the set of intended models for the application's terminology. In a language such as first-order logic, this is equivalent to saying that every sentence that is provable from the axioms of the ontology is also an inference that the application would make, and conversely, every inference that the application makes is provable from the axioms of the ontology. This property of an ontology allows to make the claim that any inferences drawn by the enterprise application using the ontology are faithful to the application's semantics. If an ontology is not verified, then it is possible to find sentences that the application infers based on its intended models, which are not provable from the axioms of the ontology. Consequently, automated inference cannot be used to determine that two enterprise applications can in fact be completely semantically integrated, and semantic integration requires human intervention. The discussion

[5] An interpretation consists of three parts:
1) a set of elements (known as the *domain* or universe of discourse)
2) a meaning function that associates symbols in the language with individual elements and sets of elements in the domain (intuitively this specifies what the symbols mean), and
3) a truth function that associates truth values with sentences in the language.
For an excellent introduction to logic, refer Barwise et al. (2000)

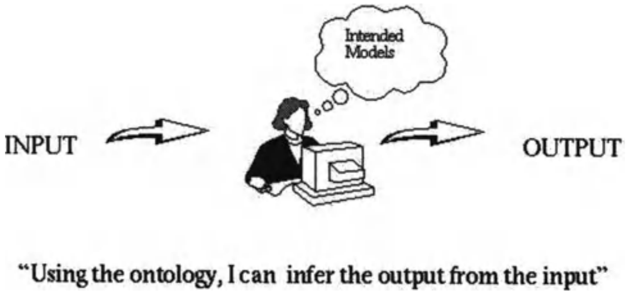


Fig. 14.3. The Ontological Stance – We can model a software application as if it were an inference system with a formal ontology, and use this ontology to predict the set of sentences that the inference system decides to be satisfiable. This is 'the Ontological Stance' (Gruninger and Menzel , 2002), and is analogous to the intentional stance (Dennett , 1989), which is the strategy of interpreting the behaviour of an entity by treating it as if it were a rational agent which performs activities in accordance with some set of intentional constraints, such as goals and obligations.

in Gruninger and Uschold (2002) considers in some detail the challenges that must be addressed in achieving semantic integration with unverified ontologies.

We can use an analogy from physics to illustrate the relationship between the ontology, the intended models, and the domain in which the enterprise application is operating. Physicists use various classes of differential equations to model different phenomena. However, they do not use ordinary linear differential equations to model heat diffusion and they do not use second-order partial differential equations to model the kinematics of springs. If physicists wish to model some phenomena using a class of differential equations, they can use the equations to predict the behaviour of the physical system; if the predictions are falsified by observations, then we have an inappropriate set of equations. Similarly, in this case, one may use some class of intended models to predict the inferences that an enterprise application makes; if there is no physical scenario in the domain that corresponds to these inferences then intuitively, the set of intended models is inappropriate.

It is important to note that this characterization of semantic integration is independent of the scope of the ontologies. Any given task may require the

use of only a portion of the ontologies shared by two enterprise applications. If that relevant portion of the ontologies supports only the intended models, then semantic integration or reusability is not impaired. For example, two applications may have different ontologies for purchasing products, but if the applications are only communicating activities and temporal constraints for scheduling their activities, then any disagreements about purchasing concepts are not relevant. In such a case, the applications can still be semantically integrated or reused with respect to the scheduling task even though they have only partially overlapping sets of concepts in their ontologies.

14.3.2 Inference

Enterprise analysis requires inference on enterprise models, including deduction, consistency checking, and abduction (hypothetical reasoning). If the semantics of the ontology is informal, then this inference must be done by an engineer or consultant and cannot be automated. However, if this inference is to be automated, then the ontology must have formal semantics. Even if the analysis is not automated, it cannot be guaranteed that two engineers will agree on the verification or validation of an enterprise model, unless the ontology has a formal semantics.

14.3.3 Implementing Enterprise Models

The dream of model-driven enterprise operations rests on the implementation of an enterprise ontology within the enterprise's information systems. The challenge of evaluating such an implementation leads to the verification of ontology implementation, which determines whether the enterprise application is correct with respect to the ontology. Without a formal semantics for the ontology, the verification of an implemented enterprise model is difficult to evaluate. However, with a formally axiomatised ontology, one can use the Ontological Stance (Fig. 14.3) as the basis for such verification. If an enterprise application is described as an inference system, then any sentence that is satisfiable by the ontology must be decided to be satisfiable by the application.

14.3.4 What is an Enterprise?

The discussion to this point has focused on the verification of an ontology, but if one carries the analogy to software engineering, then one must also address the *validation* of the ontology. In particular, does an enterprise modelling ontology have sufficient expressiveness to axiomatise the necessary enterprise modelling constructs? What range of concepts is needed for enterprise modelling?

To identify the scope of ontologies required for enterprise modelling, the first need is for a characterization of an enterprise. If an enterprise is defined as

a set of sentences whose lexicon consists of terminology whose semantics are specified by the ontologies, then the following classes of sentences would intuitively need to be definable in any validated enterprise ontology:

- Definitions of activities performed within the enterprise;
- Resource constraints for the enterprise;
- Organizational constraints for the enterprise, including constraints among organizational roles, positions, and agents within the enterprise, such as commitments, obligations, and responsibilities;
- Enterprise goals and policies (e.g. "all deliveries must be made within 24 hours of placing the order" and "when an order is made, a copy is sent to the regional office");
- Product constraints for the enterprise, including product design requirements, quality constraints, and product standards;
- Service constraints for the enterprise;
- External constraints on the enterprise defining the external environment of the enterprise, dealing with customers, markets, suppliers, and competitors. External constraints also include the definitions of the activities performed by agents external to the enterprise (e.g. suppliers, subcontractors), but whose effects are required by the activities within the enterprise.

In addition to these classes of sentences, there are several basic relationships that need to be captured:

- Customers have goals and requirements that they assign to the enterprise to achieve;
- There are two primary classes of goals, related to products and services;
- Enterprise goals are generated from the commitment to achieve customer goals and satisfy customer requirements;
- There must be ways of assigning goals of the entire enterprise to agents within the enterprise;
- If the enterprise cannot achieve a particular goal, external agents (such as suppliers or partners) would have to be assigned to achieve that goal;
- The enterprise must be able to perform those activities whose effects achieve the enterprise's goals;
- Resources are required by enterprise activities.

Of course, a truly comprehensive characterization would be provided by an enterprise ontology itself, but this intuitive characterization will later be used to evaluate existing enterprise ontologies.

14.4 Languages for Enterprise Modelling

There are many languages for expressing ontologies – and their semantic properties vary in important ways. They may be based on different underlying paradigms, they support different levels of expressiveness and their formal

properties may differ. Among the variety of languages being used today to specify ontologies, the four described in this section are the most widely used.

14.4.1 UML

The primary application of UML (Fowler and Scott , 1999) for ontology design is in the specification of class diagrams for object-oriented software. However, UML does not have a clearly specified declarative semantics, so that it is not possible to determine whether an ontology is consistent, or to determine the correctness of an implementation of the ontology. Semantic integration in such cases becomes a subjective exercise, validated only by the opinions of the human designers involved in the integration effort. More recently, UML has been supplemented with the Object Constraint Language (OCL) (Warmer and Kleppe , 1999) that is closer to offering a semantics similar to a restriction of first-order logic, and there is some research (Cranefield and Purvis , 1999) on the suitability of OCL for more rigorous ontology specification.

An additional drawback of UML as an ontology specification language is that it contains several implicit ontological commitments, particularly in respect to activity concepts. This makes it difficult to use UML to integrate different process-related enterprise applications (which may require the use of activity terminology whose semantics may not be equivalent to the intended semantics of the corresponding UML concepts).

On the other hand, UML is closer than more logic-oriented approaches to the programming languages in which enterprise applications are implemented. If there is agreement on the informal semantics of the UML-based ontology, then the verification of the implementation with respect to the ontology may be easier.

14.4.2 EXPRESS

EXPRESS (Schenk and Wilson, 1994) was initially designed to support information modelling, particularly the information required to design, build, and maintain products. Although EXPRESS generalizes earlier approaches such as IDEF1X (Menzel , 1997), the major drawback for specifying ontologies for semantic integration is that EXPRESS does not have a clear declarative semantics. This makes it difficult to verify ontologies that use EXPRESS, and also makes it difficult to determine the consistency of semantic mappings between ontologies. There are also no automated inference tools capable of reasoning with EXPRESS beyond checking for data integrity constraints.

The EXPRESS language has been accepted as an international standard (ISO10303 , 1994) and is widely used by other ISO standards, particularly the STEP standard for product data exchange.

14.4.3 DAML+OIL

The Darpa Agent Markup Language (DAML) (Hendler and McGuinness , 2001) is based on description logic (McGuinness and Patel-Schneider , 1998; Broekstra et al. , 2000), which is a specialized language that originated in the KL-ONE system of Brachman and Schmolze (1981). Description logic is a variation of first-order logic that arises from restrictions to support reasoning within class hierarchies; they also restrict first-order logic by omitting constructs that lead to undecidability of inference within the language. The Ontology Inference Layer (OIL) (Fensel et al. , 2001) is a language that extends previous frame-based languages (such as OKBC (Chaudri et al. , 1998)) with a richer set of modelling primitives. These two efforts have been merged to create DAML+OIL (DAML , 2001)

What distinguishes the DAML+OIL language from the other ontology specification languages is that it has been primarily designed for the Semantic Web (Broekstra et al. , 2000), and is intended to be compatible with emerging web standards such as RDFS (Brickley and Guha , 2000) in order to make it easier to use ontologies consistently across the web. It is currently being proposed as a standard within W3C.

Ontologies to support the semantic web are being developed using the DAML language [6]. The most important of these ontologies is the DAML-S (Milpath , 2001), which is an upper ontology for services that includes concepts for profiles, processes, and time. In this context, 'services' refer to Web sites that do not merely provide static information but allow one to effect some action or change in the world, such as the sale of a product or the control of a physical device. Thus, the DAML-S ontology must support automatic web service discovery, invocation, composition, and interoperation.

14.4.4 KIF

The Knowledge Interchange Format (KIF) (in Genesereth and Fikes (1992); Hayes and Menzel (2001)) and Conceptual Graphs (CG, in Sowa (2000)) are languages designed to support the interchange of knowledge among heterogeneous computer systems[7]. KIF includes a core language that has the expressiveness of first-order logic; its syntax and semantics are those of traditional first-order logic. Most recently, this has been extended to include extensions that allow certain formulae of infinite length (known as infinitary logic), sorted formulae for the specification of class hierarchies, and the specification of the meta-theory[8] of KIF within the language itself.

[6] A library of approximately 160 ontologies is available on-line at http://www.daml.org/ontologies/

[7] Although defined separately, both KIF and CG have equivalent expressiveness, and are being standardized together within the International Standards Organization

[8] A note to the reader: what is usually referred to in the Enterprise modelling literature as a 'model' is called in mathematical logic a 'theory'. Similarly, the concept

Several inference tools are available for reasoning with KIF/CG (such as the SNARK theorem prover (Stickel et al. , 1994), although these have had limited use outside of the academic community.

14.5 Ontologies for Enterprise Modelling

Enterprise modelling ontologies are distinguished by their scope and the central role of integrating multiple ontologies. The ontologies must be able to represent concepts in the domains of activities, time, resources, products, services, organization, goals, and policies. Furthermore, these must be integrated in order to support reasoning that requires the use of multiple ontologies and to support interoperability among tools using different ontologies. For example, the notion of *manufacturability* requires reasoning about the product properties, preconditions and effects of activities and the capabilities of resources.

14.5.1 Edinburgh Enterprise Ontology

The Enterprise Project at the University of Edinburgh (Uschold et al. , 1997) supported an environment for integrating methods and tools for capturing and analysing key aspects of an enterprise, based on an ontology for enterprise modelling.

The Edinburgh Enterprise Ontology (EEO) has five top-level classes for integrating the various aspects of an enterprise (Activities and Processes, Time, Organization, Strategy and Marketing) for integrating the various aspects of an enterprise (refer Table 14.1).

The Activities and Processes concepts define the activities and resources in the enterprise. The Organization concepts cover the organizational constraints for the enterprise. Goals, policies, and their relationship to the activities performed by the enterprise and its agents are covered by the Strategy concepts. The Marketing concepts cover the constraints that characterize the external environment of the enterprise including the relevant relationships between an enterprise, its customers, suppliers and partners. On the other hand, the Enterprise Ontology lacks a characterization of products and services.

The EEO is semi-formal – it provides a glossary of terms expressed in a restricted and structured form of natural language supplemented with a few formal axioms using KIF and Ontolingua (Chaudri et al. , 1998). As such, EEO is not a verified ontology.

Lloyd's Register has used the EEO for more effective modelling and re-engineering of business processes for strategic planning. IBM UK intends to

---

of a meta-model in Enterprise Modelling (and in most engineering disciplines) is mathematically speaking a meta-theory. Considering an enterprise model as a theory allows people and machines to make no inferences from the enterprise model that were not intended (and to be able to make all intended inferences).

Table 14.1. Concepts in the Edinburgh Enterprise Ontology

Activity	Activity Specification, Execute, Executed Activity Specification, T-Begin, T-End, Pre-Conditions, Effect, Doer, Sub-Activity, Authority, Activity Owner, Event, Plan, Sub-Plan, Planning, Process Specification, Capability, Skill, Resource, Resource Allocation, Resource Substitute.
Organisation	Person, Machine, Corporation, Partnership, Partner, Legal Organisation Entity, Organisational Unit, Manage, Delegate, Management Link, Legal Ownership, Non-Legal Ownership, Ownership, Owner, Asset, Stakeholder, Employment Contract, Share, Share Holder.
Strategy	Purpose, Hold Purpose, Intended Purpose, Strategic Purpose, Objective, vision, Mission, Goal, Help Achieve, Strategy, Strategic Planning, Strategic Action, Decision, Assumption, Critical Assumption, Non-Critical Assumption, Influence Factor, Critical Influence Factor, Non-Critical Influence Factor, Critical Success Factor, Risk.
Marketing	Sale, Potential Sale, For Sale, Sale Offer, Vendor, Actual Customer, Potential Customer, Customer, Reseller, Product, Asking Price, Sale Price, Market, Segmentation Variable, Market Segment, Market Research, Brand Image, Feature, Need, Market Need, Promotion, Competitor.
Time	Time Line, Time Interval, Time Point

exploit the Enterprise Ontology in modelling its own internal organization as well as providing technical input via its BSDM (Business Systems Development Method) business modelling method. The Enterprise Ontology is an ongoing source of inspiration for projects, both academic and commercial that require models of concepts in this domain. To the author's knowledge, the EEO is never imported or translated into a target language in full. Rather, it is perused and picked over for ideas and concepts that may be useful in the new context.

14.5.2 TOVE

The TOVE (TOronto Virtual Enterprise) project (Gruninger and Fox , 1998; Gruninger , 1997) has created an integrated suite of ontologies to support enterprise engineering. Since this suite aims to be a shared terminology for the enterprise that every application can jointly understand and use, the ontologies span knowledge of activity, time, and causality (Fox et al. , 1995; Fadel et al. , 1994; Kim and Fox , 1995; Tham et al. , 1994).
The TOVE ontologies were developed in cooperation with several companies and have been applied to the design and analysis of enterprise models within supply chain management (SCM), project management, and business process engineering. In particular Atefi (1997) discusses the application of the

TOVE ontologies to the analysis of customer relationship management processes within IBM Canada. In other work, the ontologies were used to model the supply chain of BHP Steel (Australia) and assist in the construction of management scenarios.
Figure 14.4 shows the suite of TOVE ontologies. The suite is divided into three groups: Core, Derivative, and Enterprise ontologies.

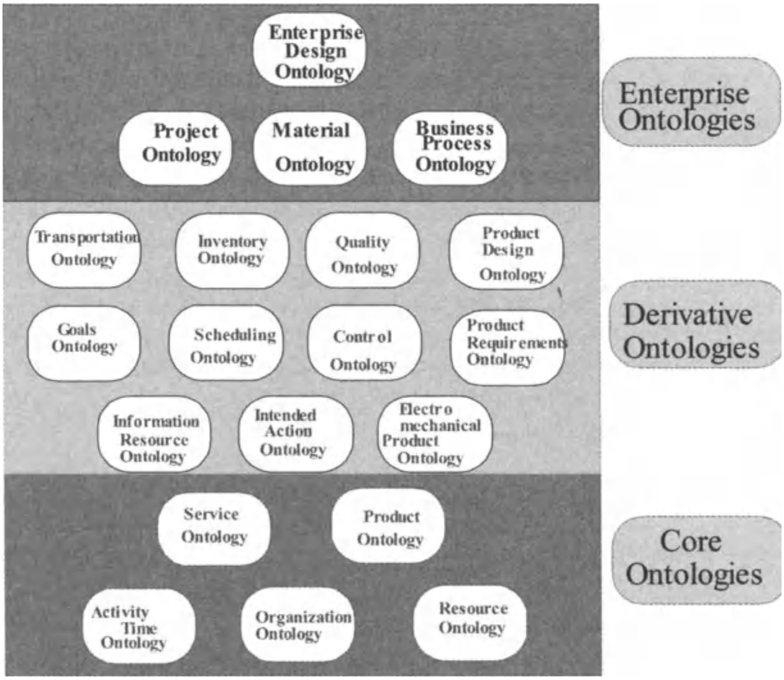


Fig. 14.4. : TOVE Ontologies

The Core ontologies capture the generic characteristics of enterprises.
The Derivative ontologies are specializations of various classes within some Core ontologies. For example, the concept of "goal" is defined in the (core) Organization Ontology, while different classes of goals (such as purchase orders and deadlines) are defined in the (derivative) Goals Ontology. An ontology may also be derivative of multiple core ontologies. For example, the Scheduling Ontology axiomatises different classes of plan and schedule activities, as well as resource and temporal constraints; it is therefore derivative of both the Activity/Time and Resource Ontologies.
There are some problems with the integration of these various ontologies within TOVE, particularly in regards to the Product Ontology. The primary

motivation for the TOVE Product Ontology was to support collaborative design. It must therefore be able to represent an evolving and incomplete design for a product, as well as represent the requirements that the product must satisfy. It must also capture the design rationale for various features and parameters of the product. However, TOVE lacks an adequate integration of the Product Ontology with the other ontologies through the following problem: Given a design for a product, how can it be manufactured? That is, what activities are required to manufacture a product with the properties specified in the design and what resources and organisational constraints are required to support these activities?

The Enterprise ontologies are used to define classes of enterprises. The Enterprise Design Ontology defines the template used to model any enterprise; as such, there is a close relationship to the informal definition of an enterprise. The various Enterprise ontologies define classes of processes, resources, products, services, and organization constraints used to define a particular class of enterprises. For example, the Material Flow ontology axiomatises the sets of processes and constraints that define supply chain enterprises, the Project ontology captures the constraints of one-of-a-kind manufacturers such as construction and ship-building, and the Business Process ontology addresses service-based enterprises. This approach is intended to support reusability and benchmarking, by identifying those constraints that are shared among different enterprises.

The TOVE ontologies are axiomatised using KIF and implemented using Prolog. Implementations of TOVE ontologies are used to analyse enterprise models in what are referred to as advisors, which are encapsulations of the theories required to reason about alternative enterprise designs (Gruninger and Fox , 1994). Advisors have included activity-based costing, quality, time-based competition, and process integration.

14.5.3 ENV 12204

ENV12204 (ENV12204 , 1995; Kosanke and Nell , 1997) describes a set of twelve modelling constructs that define the basic language for modelling enterprise operations (Fig. 14.5). In comparison to the intuitive definition of an enterprise, ENV 12204 provides adequate coverage for enterprise modelling concepts. The primary drawback of ENV 12204 is that it has only an implicit semantics expressed in natural language, and does not have an underlying ontology specification language.

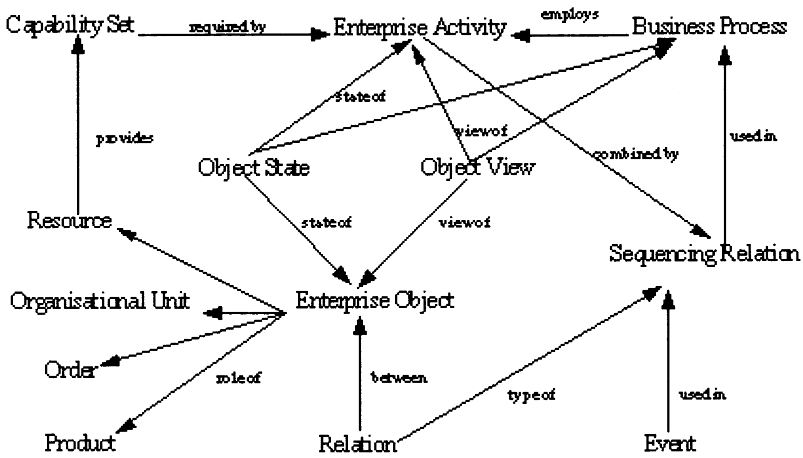


Fig. 14.5. ENV12204 enterprise modelling constructs.

14.6 Ontologies for Sets of Enterprise Modelling Concepts

Enterprise modelling ontologies explicitly construct an integrated set of smaller modules in order to capture the entire range of enterprise concepts. There are also several efforts underway within academia, industry, and government that are focused on designing ontologies for more restricted sets of enterprise concepts, such as processes, resource, and products.

14.6.1 PSL

The Process Specification Language (PSL) (Menzel and Gruninger , 2001; Schlenoff et al., 1999; Cutting-Decelle et al. , 2000) has been designed to facilitate correct and complete exchange of process information among manufacturing and business software systems. Included in these applications are scheduling, process modelling, process planning, production planning, simulation, project management, workflow, and business process reengineering.
The PSL Ontology is organized into PSL-Core and a partially ordered set of extensions. All axioms are first-order sentences, and are written in KIF . There are two types of extensions within PSL – core theories and definitional extensions. Core theories introduce and axiomatise new relations and functions that are primitive, whereas all terminology introduced in a definitional extension

have conservative definitions using the terminology of the core theories. Thus, definitional extensions do not add new expressive power to PSL-Core.

The purpose of PSL-Core is to axiomatise a set of intuitive semantic primitives that is adequate for describing the fundamental concepts of manufacturing processes[9]. Consequently, this characterization of basic processes makes few assumptions about their nature beyond what is needed for describing those processes, and the Core is therefore rather weak in terms of logical expressiveness. In particular, PSL-Core is not strong enough to provide definitions of the many auxiliary notions that become necessary to describe all intuitions about manufacturing processes.

To supplement the concepts of PSL-Core, the ontology includes a set of extensions that introduce new terminology. Any PSL extension provides the logical expressiveness to axiomatise intuitions involving concepts that are not explicitly specified in PSL-Core. All extensions within PSL are consistent extensions of PSL-Core, and may be consistent extensions of other PSL extensions. However, not all extensions within PSL need be mutually consistent. In addition, the core theories need not be conservative extensions of other core theories.

The definitional extensions are grouped into parts according to the core theories that are required for their definitions. Figure 14.5 gives an overview of these groups together with example concepts that are defined in the extensions. The definitional extensions in a group contain definitions that are conservative with respect to the specified core theories; for example, all concepts in the Temporal and State Extensions have conservative definitions with respect to both the Complex Activities and Discrete States theories.

PSL is a project within Joint Working Group 8 of Sub-committee 4 (*Industrial data*) and Sub-committee 5 (*Manufacturing integration*) of Technical committee ISO TC 184, (*Industrial automation systems and integration*). Part 1 of the standard has been accepted as a Committee Draft citepISO00. All theories within the PSL Ontology that are currently being standardized have been verified with respect to the intended semantics of their terminology.

14.6.2 STEP and MANDATE

STEP (ISO10303 , 1994) has been standardized within the International Standards Organization to support interoperability among manufacturing product software applications (such as CAD systems and process planning software) throughout the entire product life cycle. STEP provides standard data definitions for geometry (wire frame, surfaces and solid models), product identification, product structure, configuration and change management, materials, finite element analysis data, drafting, visual presentation, tolerances, kinematics, electrical properties and process plans. STEP is currently being implemented in the aerospace, automotive, shipbuilding, building design and electronics industries.

[9] The axioms of PSL-Core were directly incorporated from earlier work with the Process Interchange Format (PIF) (Lee et al. , 1998)

Table 14.2. Definitional extensions of PSL.

Definitional Extensions	**Core Theories**	**Example Concepts**
Activity Extensions	Complex Activities	Deterministic / Non-deterministic activities Concurrent activities Partially ordered activities
Temporal and State Extensions	Complex Activities, Discrete States	Preconditions Effects Conditional activities Triggered activities
Activity Ordering and Duration Extensions	Sub-activity Occurrence Ordering, Iterated Occurrence Ordering, Duration	Complex sequences and branching Iterated activities Duration-based constraints
Resource Role Extensions	Resource Requirements	Reusable, consumable, renewable, deteriorating resources

MANDATE (ISO15531 , 1999) is also being standardized within Joint Working Group 8 of Sub-committee 4 (*Industrial data*) and Sub-committee 5 (*Manufacturing integration*) of Technical committee ISO TC 184, (*Industrial automation systems and integration*). MANDATE is primarily concerned with manufacturing resource data, including an informal ontology of time.
Both STEP and MANDATE are specified in EXPRESS; consequently, they cannot be verified with respect to their intended semantics.

14.7 Challenge Problems for Enterprise Modelling

This chapter concludes with five challenge problems to motivate future research within enterprise modelling.

14.7.1 Ontologies for Enterprise Modelling

Two ontologies for enterprise modelling have been constructed in the past decade – TOVE and the Edinburgh Enterprise Ontology. Although there is considerable overlap in the set of concepts in each of these ontologies, no effort has been made to merge or align them. Such an alignment could at least form the basis for a formalization of the concepts informally described in

ENV 12204. More ambitiously, an alignment of these two enterprise modelling ontologies could be used as the basis for standardization of a wide range of enterprise concepts.
An initial step in this direction is the Unified Enterprise Modelling Language (UEML), a new project whose goal is to provide a common language suited for enterprise modelling (Kosanke and Nell , 1997). It is intended to provide business users with a standard interface to software for enterprise modelling, analysis, and simulation. It also aims to provide a neutral language for enterprise model exchange.

14.7.2 Implementing Ontologies

There is very little work being done on the implementation of ontologies within enterprise applications. In fact, many of the ontologies for enterprise integration are designed post-hoc by extracting the ontology implicit within existing enterprise applications. However, as ontologies are extended to new domains (particularly for organizational constraints and electronic commerce), new applications will be implemented directly from the ontologies. Thus, a methodology for the evaluation of such implementations is needed.

14.7.3 Ontology Reuse

Although ontologies came to prominence within artificial intelligence via the DARPA program for Sharable and Reusable Knowledge Bases (Neches et al. , 1991; Gruber , 1993), there is still limited reuse and sharing of ontologies. It is difficult to determine why this is the case ((Uschold et al. , 1998; Pinto , 1999; Goldstein and Esterline , 1995). The Ontolingua ontology library at the Stanford University Knowledge Systems Laboratory contains almost 100 ontologies (http://www.ksl.stanford.edu/software/ontolingua), but there are limited links among most of them. Within the context of semantic integration, this becomes the problem of how enterprise applications determine that they have overlapping sets of concepts and how could they possibly share the semantics of their terminology.
The challenge here is to build an ontology for enterprise modelling that integrates existing ontologies for process, product, resource, and organization. Such ontologies often have overlapping concepts, and these may cause problems with reuse. For example, the Standard for the Exchange of Product data (STEP - ISO10303 (1994)) was designed for product modelling and the Process Specification Language (PSL) (Schlenoff et al., 1999) was designed for process modelling. However, both ontologies contain the concept *process-plan*, which is the sequence of activities that must be performed to manufacture a product according to its design specifications. Unfortunately, this concept is defined very differently in the two ontologies, preventing easy reuse between them.

Enterprise (life-cycle) architecture frameworks, such as GERAM (refer Chapter **3**) may be used to organise ontological theories and define the scope of ontology development for enterprise modelling purposes. The 'generic enterprise models' column of GERAM may be populated with ontological theories. At the moment this column is populated by so-called meta-models. Mathematically speaking this kind of meta-model is a weak ontological theory represented in the form of a meta-schema, defining the concepts and relations between them, but without expressing inference rules or semantic integrity constraints (with the exception of very simple constraints). Thus a meta-schema can be used to design a database to store enterprise models, but can not be used to perform deductions on these models One way to develop an enterprise ontology is to define a meta-schema and then add inference rules and semantic integrity constraints to the concepts defined in such a meta-schema.

14.7.4 Ontology Extension

How could generic ontologies be extended to more domain-specific ones? This problem appears in the distinction between Core and Derivative ontologies in TOVE, but there is no coherent methodology for ontology extension. Many ontologies originate as domain ontologies, within different applications' and scientific disciplines' ontologies ((Ashburner, 2000; Cohn , 2001; Dalianis and Persson , 1997; Smith and Becker , 1997)). It may be argued that there are few domain concepts in common between physics and logistics and hence little reuse may exist between ontologies for these domains. However, such domain ontologies often use very similar generic concepts (for example, both ontologies may contain an ontology of time). The challenge of reuse and sharing is then equivalent to the task of identifying the generic concepts that are implicit within a domain ontology. In fact, the goal of the Standard Upper Ontology project (Pease , 2001) is to a define generic ontology that more domain-specific ontologies can reuse in this way.

14.7.5 Enterprise in a Test Tube

There are many issues within enterprise integration that can only be resolved through empirical approaches. There is a need to establish an academic and industrial testbed (which will be refer to as an 'Enterprise In a Test Tube' (ETT)) that consists of multiple enterprise applications and ontologies. Using this environment, participants would carry out experiments to test, compare, and validate various theories about enterprise design and reengineering.
One problem is that enterprise design knowledge is currently descriptive and ad-hoc. It is a collection of heuristics that are not applicable in all circumstances. Therefore, it is desirable to define a theory of enterprise design by discovering its underlying principles. It has to be understood why different approaches and techniques work for certain enterprises and why they fail for

other enterprises. There is a need for a distillation of the principles for enterprise design implicit within the heuristics, and the formalisation of these principles as logical theories. Once this is accomplished, various enterprise design theories could be tested, compared and validated.
The use of integrated ontologies allows the flexible configuration of enterprise models and operating scenarios for problems. For example, operating strategies within an enteprise (such as quality problem response, production strategies and inventory management policies) would be explicitly represented in the enterprise model, supporting a 'plug-and-play' approach to the incorporation and change of constraints in a problem specification. Hypotheses for enterprise design heuristics are expressed as queries that can be deduced from the axioms of the ontologies and theories.
The ETT should also support new ways of building enterprise models, particularly in the acquisition and validation of an enterprise model. It should support the capability of reconciling different enterprise designs that may arise during the acquisition process. Model acquisition must therefore be able to handle incomplete and inconsistent information, as well as being able to modify or augment a model when things do not work. It should use partial enterprise models, combine these partial models into an integrated model of the entire enterprise, and support the iterative refinement / elaboration and definition of the enterprise model, 'filling in' pieces of incomplete models.
To be effective, the ETT must also support reusability by providing a repository for various enterprise models, including previous problems and their solutions[10]. ETT must provide the capability of dynamically constructing and modifying models, so that new models can be created from existing models by reconfiguring them to adapt to a given problem.

References

Ashburner, M.(2000) Gene ontology: Tool for the unification of biology. *Nature Genetics* 25:25-29.

Atefi, K.(1997) Formalization of Business Process Re-engineering Heuristics, M.A.Sc. Thesis, Mechanical and Industrial Engineering Dept., University of Toronto, 1997.

Barwise, J., Etchemendy, J., Allwein, G., Barker-Plummer, D.(2000) Language, Proof, and Logic. Seven Bridges Press.

Brachman, ? and Schmolze, ?.(1981) Reference Missing

Brickley, D. and Guha, V.R.(2000) Resource Description Framework Schema Specification 1.0. W3C Candidate Recommendation. (http://www.w3.org/TR/rdf-schema)

Broekstra, J., Klein, M., Fensel, D., and Horrocks, I.(2000) Adding formal semantics to the Web: Building on top of RDF Schema. *Proceedings of the ECDL 2000 Workshop on the Semantic Web.*

[10] e.g. in the form of enterprise modelling *patterns*.

Chaudri, V.K., Farquhar, A.,Fikes, R., Karp, P.D., and Rice, J.P.(1998) OKBC: A programmatic foundation for knowledge base interoperability, *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisoconsin, 1998

Chaudhri, V. K., Farquhar, A., Fikes, R., Karp, P.D.and Rice, J. P.(1998) Open Knowledge Base Connectivity 2.0. Knowledge Systems Laboratory, 1998.

Ciocoiu, M., Gruninger M., and Nau, D.(2001) Ontologies for integrating engineering applications, *Journal of Computing and Information Science in Engineering*, 1:45-60.

Cohn, A. (2001)Formalizing biospatial knowledge, *Formal Ontology in Information Systems* 2001, Ogunquit, Maine.

Cover, R. (1998)XML and Semantic Transparency, The XML Cover Pages http://www.oasis-open.org/cover/xmlAndSemantics.html

Cranefield, S. and Purvis, M.(1999) UML as an ontology modelling language. In *Proceedings of the Workshop on Intelligent Information Integration*, Sixteenth International Joint Conference on Artificial Intelligence.

Cutting-Decelle, A.F., Anumba, C.J., Baldwin, A.N. and Gruninger, M. (2000)Towards a unified specification of construction process information: The PSL approach, in *Product and Process Modelling in Building and Construction*, Steiger-Garcao and Scherer (eds), 199-207.

DAML.(2001) - Reference Missing !!

Daliannis, H. and Persson, F.(1997) Reuse of an ontology in an electrical distribution network domain, *Ontological Engineering*, AAAI-97 Spring Symposium Series, Stanford.

Dennett, D.(1989) The Intentional Stance. MIT Press.

ENV12204: Advanced Manufacturing Technology – Systems Architecture – Framework for Enterprise Modelling. CEN TC 310/WG1, 1995.

Fadel, F., Fox, M.S., and Gruninger, M.(1994) A resource ontology for enterprise modelling. Third Workshop on Enabling Technologies-Infrastructures for Collaborative Enterprises, (West Virginia University 1994), pp. 117-128.

Fensel, D., van Harmelen, F., Horrocks, I., McGuinness, D. L., and Patel-Schneider, P. F. (2001) OIL: An ontology infrastructure for the semantic web. IEEE Intelligent Systems, 16(2):38–44. http://citeseer.nj.nec.com/fensel01oil.html

Fowler, M. and Scott, K.(1999) UML Distilled: A Brief Guide to the Standard Object Modelling Language. Addison-Wesley.

Fox, M.S., Barbuceanu, M., Gruninger, M.(1995) An Organisation Ontology for Enterprise Modelling: Preliminary Concepts for Linking Structure and Behaviour, *Computers in Industry*, Vol. 29, pp. 123-134.

Genesereth, M.R. and Fikes, R.(1992) Knowledge Interchange Format 3.0. Technical Report KSL-92-01, Knowledge Systems Laboratory, Stanford University.

Goldstein, D., and Esterline, A.(1995) Methods for Building Sharable Ontologies, Proceedings of the IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing, Menlo Park CA, USA: AAAI Press, 1995.

Gruber, T.R.(1993) A translation approach to portable ontology specifications *Knowledge Acquisition* 5:199-220.

Gruninger, M. and Menzel, C.(2002) Process Specification Language: Principles and Applications, to appear in *AI Magazine*, 2002.

Gruninger, M. and Uschold, M.(2002) Ontologies and semantic integration, to appear in *Software Agents for the Warfighter*. Information Technology Assessment Consortium.

Gruninger, M., and Fox, M.S.(1998) Enterprise Modelling, *AI Magazine*, 19:109-121 (Fall 1998), AAAI Press.

Gruninger, M.(1996) Designing Generic Ontologies, *Workshop on Ontological Engineering*, European Conference on Artificial Intelligence 1996, Budapest.

Gruninger, M.(1997) Ontologies for Enterprise Engineering, *Enterprise Engineering and Integration: Building International Consensus*, Springer-Verlag.

Gruninger, M., and Fox, M.S.(1994) The Role of Competency Questions in Enterprise Engineering*Benchmarking - Theory and Practice*. Chapman Press.

Guarino, N., Carrara, M., and Giaretta, P. (1994) An Ontology of Meta-Level Categories. In E. Sandewall and P. Torasso (eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94)*. Morgan Kaufmann, San Mateo, CA: 270-280.

Guarino, N. and Welty, C.(2000) Identity, unity, and individuality: Towards a formal toolkit for ontological analysis. *Proceedings of the Fourteenth European Conference on Artificial Intelligence*, Berlin.

Hayes, P. and Menzel, C.(2001) A semantics for the Knowledge Interchange Format, *Workshop on the IEEE Standard Upper Ontology*, IJCAI 2001, Seattle.

Hendler, J. and McGuinness, D.(2001) The DARPA Agent Markup Language. *IEEE Intelligent Systems*, January 2001.

ISO (1994) ISO10303:Industrial Systems and Automation – Product Data, 1994.

ISO(1999) ISO15531: Industrial Systems and Automation – Industrial Manufacturing Management Data, 1999

ISO/TC184/SC5/WG1(1999) Annex A: GERAM. In *ISO/DIS 15704: Industrial automation systems - Requirements for enterprise-reference architectures and methodologies*, 1999

ISO(2000). ISO18629:Industrial Systems and Automation – Process Specification Language,2000

Kim, H. and Fox, M.S.(1995) An Ontology of Quality for Enterprise Modelling, Fourth Workshop on Enabling Technologies-Infrastructures for Collaborative Enterprises, (West Virginia University 1995).

Kosanke, K. and Nell, J.(eds.) (1997) *Enterprise Engineering and Integration: Building International Consensus*. Springer-Verlag.

Lee, J., Gruninger, M., Jin, Y., Malone, T., Tate, A., Yost, G.(1998) The PIF Process Interchange Format and Framework, *Knowledge Engineering Review*, 2:1-30.

McGuinness, D.L. and Patel-Schneider, P.F.(1998) Usability issues in description logic systems, *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin.

Milpath, S., Son, T.C., and Zeng, H.(2001) Semantic web service, *IEEE Intelligent Systems* 16:46-53.

Menzel, M.(1997) Modelling method ontologies: A formal foundation for enterprise model integration, *Workshop on Ontological Engineering*, AAAI-97 Spring Symposium, Stanford.

Menzel, C. and Gruninger M.(2001) A formal foundation for process modelling, *Formal Ontology in Information Systems 2001*, Ogunquit Maine.

Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T., and Swartout, W.(1991) Enabling technology for knowledge sharing, AI Magazine, Vol. 12, No. 3, 36-56.

Pease, A.(Ed.)(2001) *Proceedings of the IEEE Standard Upper Ontology Workshop*, IJCAI-2001, Seattle.

Pinto, H.S.(1999) Towards ontology reuse, *Workshop on Ontology Management*, AAAI-99, Orlando.

Schenk, D.A. and Wilson, P.R.(1994) Information Modelling the EXPRESS Way. Oxford University Press, 1994.

Schlenoff, C., Gruninger, M., Ciocoiu, M.(1999) The Essence of the Process Specification Language, *Transactions of the Society for Computer Simulation vol.16 no.4 (December 1999) pages 204-216.*

Smith,I., Cohen, P., Bradshaw,J., Greaves, M. and Holmback, H.(1998) Designing Conversation Policies using Joint Intention Theory. Proceedings of the Third International Conference on Multi Agent Systems (ICMAS-98), 3 - 7 July, 1998, Paris, France, IEEE Press, pp. 269-276.

Smith, S. and Becker, M.(1997) An ontology for constructing scheduling systems, *Ontological Engineering*, AAAI-97 Spring Symposium Series, Stanford.

Sowa, J.F Knowledge Representation: Logical, Philosophical and Computational Foundations. Brooks/Cole.

Stickel, M., Waldinger, R., Lowry, M., Pressburger, T., and Underwood, I.(1994) Deductive composition of astronomical software from subroutine libraries, *Proceedings of the Twelfth International Conference on Automated Deduction*, Nancy, France, pp.341-355.

Tham, D., Fox, M.S., and Gruninger, M.(1994) A Cost Ontology for Enterprise Modelling, Third Workshop on Enabling Technologies-Infrastructures for Collaborative Enterprises, (West Virginia University 1994).

Uschold, M. and Gruninger, M.(1996) Ontologies: Principles, Methods, and Applications, *Knowledge Engineering Review*, 1:96-137.

Uschold, M., King, M., Moralee, S., Zorgios, Y.(1997) The Enterprise Ontology, Knowledge Engineering Review

Uschold,M., Healy,M., Williamson,K., Clark,P., Woods, S.(1998) Ontology Reuse and Application. In *Proceedings of the International Conference on Formal Ontology and Information Systems - FOIS'98 (Frontiers in AI and Applications v46)*, pages 179-192, Ed: N. Guarino, Amsterdam:IOS Press, 1998.

Warmer, J. and Kleppe A.(1999) The Object Constraint Language: Precise Modelling with UML. Addison-Wesley.

Part IV

Developing the Master Plan - Architectural Design of the Changed Enterprise

15

PRELIMINARY DESIGN : TRANSLATING REQUIREMENTS TO DESIGN SPECIFICATIONS

David Chen[1], Bruno Vallespir[1], and Guy Doumeingts[1,2]

[1] Laboratoire GRAI/LAP, University of Bordeaux I
[2] GRAISoft

15.1 Preliminary Design and General Activity of Design

Design is one of the phases of the system life cycle of GERA (Generalised Enterprise Reference Architecture) which itself is a component of GERAM[3]. Design is split into two sub-phases: preliminary design [4] and detailed design. This section presents basic concepts and principles for preliminary design. The preliminary design of manufacturing systems will be used as a running example, however, the processes and rules presented here are equally valid for any kind of complex system.
Preliminary design looks at the system in question from various points of view. These views can be categorised according to :

- the purpose of activity (mission fulfilment - i.e. service and production vs. management and control);
- the means of implementation (human vs. automated);
- physical manifestation (hardware vs. software);
- modelling views that enable the designer to express the model of the system from different aspects and through this to classify problems to be solved (thereby reducing the number of facts that need to be considered at any one time).

Several modelling views may be chosen, depending on the approach supporting the study in question. For example, the GIM/GRAI architecture and methodology uses functional, physical, decisional and informational views (Doumeingts et al., 1992). The relevance of this is that GIM/GRAI was one of the first methodologies to deal with the problems of architectural design in manufacturing systems, thus the principles in this Chapter are based on that methodology. In terms of the GERA modelling framework, the GIM/GRAI

[3] Generalised Enterprise Reference Architecture and Methodology, as described in (IFAC-IFIP Task Force , 1999)
[4] Preliminary design is also commonly known as *architectural design*

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

functional view is a high level functional view of the entire system and the physical view is a functional view of the mission fulfilment part of the system. The decisional view is a special functional view of the management and control part of the system, particularly suited to designing management functions (the system of decisions), and the information view is an ordinary information view of the entire system. This is consistent with the GERA modelling framework that defines model view *categories* (function, information, organisation and resource views), with the intention to reflect the fact that different methodologies need to fill these categories with selected special purpose views - such as decisional view, economic view, etc. While the philosophy of the authors of this Chapter is based on GIM/GRAI and their original publications use the above GIM/GRAI views, it was considered appropriate to use the generic GERA model view categories. This decision does not effect the substance of the principles below, but makes it easier for the reader to relate to the description of preliminary design principles to other parts of this book. The result of preliminary design is a set of deliverables, or outcomes, therefore architectural design information is captured in the views above. It is not necessary (in fact improbable) that every practitioner will use exactly these deliverables to structure preliminary design information, however, any of the above views of preliminary design outcomes should be able to be derived when preliminary design is deemed to be complete.

15.1.1 The Activity of Design

Generally speaking, design is about the creation of artefacts. In the engineering domain, the term 'design' means to elaborate the explicit specifications of the structure of a product or system that satisfy desired requirements under constraints.

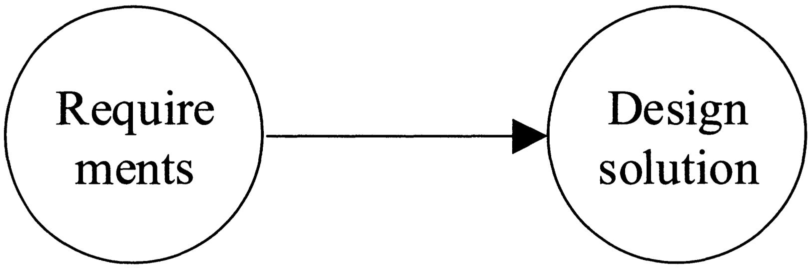


Fig. 15.1. Design as a mapping from requirements to solution.

Design can be considered as an iterative process, which manipulates knowledge on existing known artefacts (products and systems) in order to create (or specify) new artefacts satisfying a list of requirements.

Design may involve various kinds of activities, such as *search, decision-making, problem-solving, planning, selection, analysis, synthesis, evaluation*, etc. In (Eynard, 1999), two basic kinds of activities are highlighted:

- the activities of *design*, which aim at generating candidate solutions, and
- the activities of *choice*, which select one among several solutions.

Design activities increase the *variety* about the future artefact (by imagining several solutions) and the choice activities reduce it. However, both increase the *knowledge* about the future artefact (Fig. 15.2).
Designing complex systems, for example manufacturing systems, needs to use models, and various views (aspects) of the models of the system under consideration and of its sub-systems. Among all possible views (which is an open list), two of them are fundamental : functions (representing design requirements) and (resource) structure (representing the design solution).

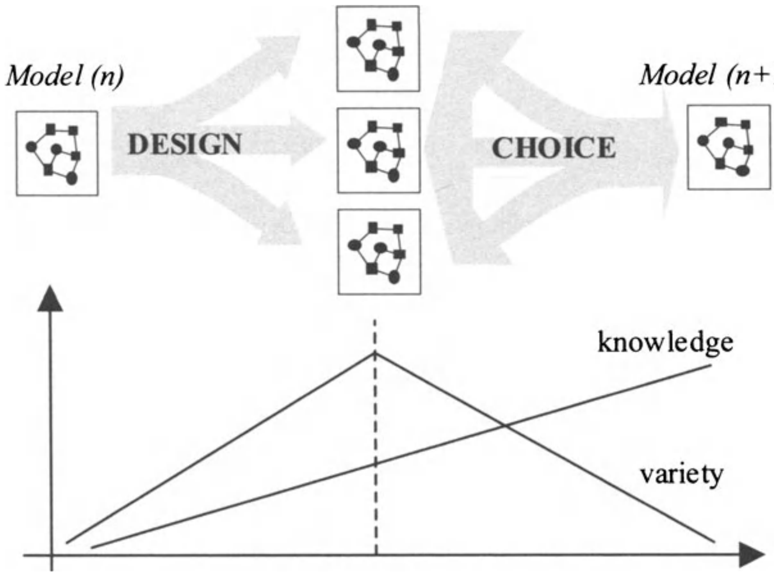


Fig. 15.2. Basic activities of design.

15.1.2 Position of Preliminary Design

Preliminary design consists in translating requirements into design specifications, which are intermediate solutions that will be further specified by detailed design.

Generally speaking, preliminary design aims at transforming required activities/processes, information/data, resource requirements and organisational requirements into three types (domains) of specifications:

- Information technology components (hardware and software);
- Manufacturing technology components (machines, robots, human operators,...);
- Organisation structure (assignment of decision making activities to people and organisational entities of the company including the notions of responsibility and authority) as illustrated in Fig. 15.3.

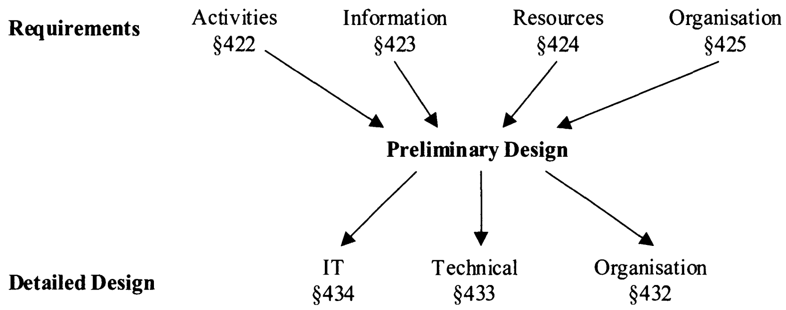


Fig. 15.3. Preliminary design phase, between requirements and detailed design[6].

At the preliminary design phase, specifications will be given at the level of 'solution types' without necessarily choosing (deciding) on the specific components of the system. For example, one may decide to implement a metal cutting function using an NC cutting machine type and specify some characteristics (speed, dimension, precision,...). This specification will be further detailed (if necessary) at the detailed design phase in order to select a commercial machine available on the market.
Several reasons justify the separation between preliminary and detailed design activities:

- Detailed design is usually carried out in a specific domain (such as information technology, manufacturing hardware, etc). For this reason, preliminary design is also called 'architectural design', determining how the system will be constructed out of various human and technological components, as well as how they interact: i.e., preliminary design is a place of 'integration'.

[6] the Chapter numbers in the figure indicate where various issues are studied in the book.

Integration is achieved if elements of a system are connected physically, and/or through information flow (ordinary information exchange and decision frameworks). Information exchange must be based on commonly understood syntax and semantics of data, and decisional information must flow without inconsistencies or contradictions between the objectives between various decisional levels.

Integration starts at a conceptual level. It means that integration starts by considering concepts of / about the system. Therefore it is necessary to make suitable abstractions of the physical system. This enables the designer to first define the future system conceptually eliminating considerations that are irrelevant from the point of view of system requirements. Furthermore, preliminary design is based on a 'global view' of the system, while detailed design focuses on local views.

Note that it is ultimately not important whether the designer first experiments with physical objects and then conceptualises their interconnections to see if their interfaces are consistent, or first conceptualises the building blocks and their interfaces and then finds or defines physical components that satisfy this conceptual model [7]. I.e. 'being integrated' is a conceptual property of the system that consists of components. For a system to be 'integrated through information flow' means that components can exchange data and interpret them in a uniform way, i.e. data exchanged between components are interpreted by the connected components of the system as the *same* information. What matters is that conceptualisation helps the designer to either verify selections of components or to actually select components based on this conceptualisation.

Which comes first, conceptualisation or the identification of potential components, depends on the style of the designer, the designer's background knowledge and experience.

This is one of the reasons why design methodologies - with the exception of routine design - are difficult or impossible to express in a generic way as 'process models', i.e. in terms of sequential steps. Any particular design project needs to consider what is the best sequence of actions suited to the design task at hand, in other words the generic functional model of the methodology needs to be translated to a particular process model of the design project. In less complex projects this may be done intuitively,

[7] in this second scenario, the designer has to be aware of the range of existing available objects and/or of the physical limitations in defining new objects which satisfy the design specifications (so that the artefact being designed may actually be built)

but in the design of complex systems it is well advised to overtly express this process model and use it as the basis of a design-project plan.

- Define and validate solution types.

  When designing a complex manufacturing system, it is pertinent to first define and validate a solution type (solution principle) with customers / users before going on to specify any technical details (for example to use an NC machine instead of a conventional machine, or to use a mainframe computer instead of a network of PCs). In this way, preliminary design enables the designer to evaluate the design according to financial constraints. It provides an opportunity to decide to continue, to stop or to re-orient the design project before spending too much time and money.

- Preliminary design allows faster iteration between evolving requirements and design than iteration between detailed design and requirements.

  The requirements that the system must satisfy cannot be completely defined at the beginning of the design. Requirements are often inconsistent, sometimes contradictory or would require an infeasible solution. While in the requirements phase there is an attempt made to 'rewrite' the requirements based on incompleteness or inconsistencies discovered at that stage, it is often not possible to make these discoveries before an architectural design is attempted. Therefore requirements must be continuously revised and completed in every stage of design - both preliminary and detailed (Pahl et al , 1996). Design activities may also be needed to discover new requirements (such as those that could not be stated before an architectural solution is at hand) or to clarify existing ones. In fact, design requirements can only be considered as complete when the design is finished (Grabovski et al, 1999). Preliminary design allows better and faster interaction with the requirements definition phase and early detection of any inconsistencies or the lack of feasibility.

  In software engineering, for example, the concept of 'user requirements' is quite different from the concept of 'system requirements'. User requirements as expressed and understood by the user / customer concentrate on what the user wants. System requirements are a translation of these into a requirements specification guiding system designers and implementors in a preferred designed system satisfying the user requirements.

  An important activity in translating user requirements to system requirements is 'orthogonalisation'. Designers need to disect user requirements and find mutually independent (thus orthogonal) base functions that underly the functionality required by the user. It is then from these orthogonal set of functions that user functionality can be built.

The reason for orthogonalisation is that user requirements not only change during the design process, but new requirements also appear during an extended operation of the system. Orthogonal design allows the system to be incrementally modified, or evolved, because the designed system is potentially capable of covering any combination of its base functions, thus any conceivable meaningful combination is a potential (albeit not actual) functionality of the resulting design.

The orthogonality principle will be revisited when dealing with the aggregation of functions below.

15.1.3 Preliminary Design and Design Principles

Preliminary design aims at specifying a future system's functionality and structure, rather than determining what commercially available components (machines, databases, software, computers,...) to use. In other words, preliminary design describes conceptually the system structure with the view of satisfying the required functions (in its general terms) within the set of constraints imposed (such as restricting design principles, financial or time constraints, risk, maintainability, etc...). Some basic concepts and principles are stated below:

- Integrated system solutions cannot be purchased 'off the shelf' on the market. An overall design solution satisfying all specific requirements of a complex system, such as a manufacturing system is non-existent. An integrated design solution must be built on the basis of elementary solutions and the state-of-the-art theories and technology;
- Design is a mapping between required properties (requirements) and the structure of the artefact (solution). However, design also includes the refinement of properties as well as the refinement of solutions;
- Knowledge about available solutions may be classified and aggregated into classes. At each stage of design, the set of models representing requirements are compared with suitable classes of solutions. Since both requirements and solutions are refined simultaneously, models may be detailed as 'sub-models' and structures as sub-structures. Appropriate solution elements are selected and then integrated. This basic process is repeated until the elementary level is reached, where components available on the market can be mapped onto requirements. Importantly, if a requirement is subdivided into two parts and a solution is subdivided into two subsystems (sub-structures) then to validate the solutions one must not only show that both requirements are satisfied, but also that the two subsystems integrated into one system will simultaneously satisfy both requirements. Thus both validation and testing must be done at subsystem and system levels (this latter is called integration testing / validation of structure);

- Design knowledge about existing system components (for example the set of manufacturing technology components and information technology components that must remain part of the solution) needs to be described. It is evident that the way of representing this knowledge will influence the way the design process is defined and performed.
- As illustrated in Fig. 15.4, awareness about- and classification of technical solutions and corresponding market-available components is prior assumed knowledge for design. This leads to another subject that must be explicitly studied. If the classification of all market-available useful components for building manufacturing systems is difficult due to the excessive amount of information, it would still be feasible to maintain such a knowledge base for a given specific, thus limited, domain such as mechanical components, software, etc.

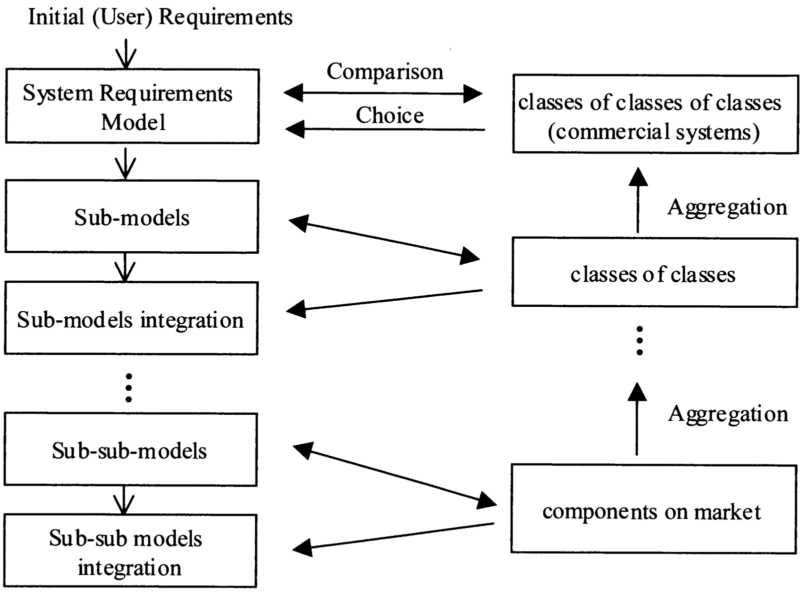


Fig. 15.4. Design as mapping and refinement.

The objective of preliminary (i.e. architectural) design (also see section **15.1.2**) is to give an orientation to design. Therefore, in this process the designer does not / should not want to specify a particular solution in detail. As soon as the designer succeeds in mapping the elements of four model views (for example) to the three domains (assignments)[8] by determining solution

[8] the four model views refer to the GIM/GRAI Function, Information, Decision, Physical and Functional (fulfilling the GERA Function, Information, Resource

principles to be used, architectural design may stop (choice of a resource or a class of resources in relation to a possible "class of available components" either implicitly known to the designer, or explicitly formalised). If this mapping fails, **refinement** would be needed to go into further detail (to clarify, decompose or revise requirements) until a solution principle is found.

15.1.3.1 The Refinement Dimension

Because design requirements cannot be completely and consistently identified when design starts, it must involve activities to identify, to revise and to refine design requirements. In real design situations, design is an evolutionary process alternating between mapping of requirements to solutions and refinement as illustrated in Fig. 15.5 (Chen et al, 2000).

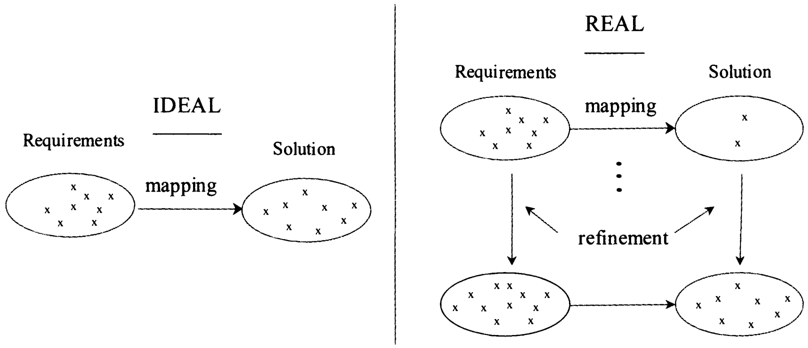


Fig. 15.5. The refinement dimension of the real design process

Refinement means to detail, to decompose, to elucidate the meaning etc. Refinement in design is necessary when design requirements cannot be correctly understood or processed by the designer. Refinement includes:

- decomposition of a required function into elementary functions so that a set of elementary components can be found and agreed on. The original function is then assembled, or aggregated, from these elements;
- definition of a precise expression of the required function, e.g. when the term used in the original requirements is too generic or too vague to find a corresponding solution, or to determine which one of possible solutions is acceptable.

---

and Organisation view categories requirements), while the three domains refer to the GIM/GRAI Organisation, Manufacturing Technology and Information technology.

Similarly to function refinement, required information elements also need to be refined to agree on inputs (and preconditions on them), and outputs (as well as post-conditions on them).

15.1.3.2 Determining Solution Principles

Solution principles are usually determined at the preliminary design phase. To choose a solution principle means to define the main orientation of a solution type without going into its technical details (to be worked out in the detail design phase once the solution principle has been validated by preliminary design). For example, to choose a relational database is to choose a solution principle; to implement using an ORACLE database management system is to choose a component from the market. The first choice is a preliminary design task while the second belongs to detail design.
Solution principles (or design principles) do not usually (and certainly do not exclusively) originate from the statement of user requirements. Design principles are generic rules and constructs that a designer can follow to produce a category of designs. Design principles are not independent of each other: they may be a shared set of principles belonging to a school of design, and may guarantee certain solution properties. E.g., one can list a set of principles that, if adhered to, ensure greater flexibility, better reconfigurability and lower implementation risk compared to the result of a design process without a coherent set of such principles.
Examples of solution principles for manufacturing system design are:

- use flexible manufacturing cell technology instead of conventional machines for a mechanical workshop design;
- use group technology based organisation (a family of products will be manufactured within one cell) instead of grouping machines according to their types;
- use a machining centre having the three functions i.e. turning, drilling and milling instead of implementing three individual machines, each performing one function;
- use radio-based local area network instead of cable-based one; use a main frame instead of PC in network;
- use MRPII management technique and not KANBAN technique for designing the production control sub-system,...
- use pressed air energy than electrical power because of explosive characteristic of product manufactured.
- etc.

The determination of a solution principle at preliminary design stage allows users and other stakeholders to validate the orientation of design solution earlier. This would reduce time and cost compared with the case where the validation is done at the detail design stage.

To reduce the number of iterations between preliminary and detail designs, explicit links between solution principles and components available on the market need to be established. It is then necessary to classify commercial products according to known solution principles.
Since end users are generally not familiar with solution principles (or are familiar only with a subset of these) it is important that a design team should be assembled that shares a common set of principles, and who can communicate design decisions both among themselves as well as to the end user (when it is necessary to explain design decisions).

15.1.3.3 Minimising Dependency of Design Decisions

The approach that seeks to minimise the number of design decisions is based on the following concepts:

- design can be considered as a decision-making process where a certain number of decisions must be taken to elaborate a solution;
- not all decisions need to be taken in preliminary design, but only those that will significantly contribute to and influence the design solution;
- decisions should be taken in a certain order of priority in order to limit their impact on the decisions taken previously.

This approach can be structured into 5 steps:

1. Establish functions required by users (e.g., a list of functional requirements for a loudspeaker)
2. Identify significant decisions that have to be made by design. These are decisions that allow the designer to specify the basic structure - i.e. without every detail (e.g. active or passive speakers are to be used?).
3. Determine decision variables and constraints. Generally speaking there is at least one variable per decision (e.g. decision variable: power source - speakers will be independently powered or not? constraint: - total power not to exceed certain limit, cabling to be simple to set up).
4. Analyse the dependency between decisions (when one variable is modified, the other decisions will be affected).
5. Regroup decisions in a certain sequence so that the dependency between decisions can be minimised.

Sometimes it is not easy to isolate one design decision from others. However, if dependencies are minimised, it will be easier to find independent *groups* of decisions. In this way, instead of looking for a sequence for individual decisions for each possible design choice, it is possible to identify a sequence of groups of decisions. Decisions within one group may then be taken concurrently.

15.2 Main Characteristics of Preliminary Design

15.2.1 Clarification of Requirements

It is difficult to define exactly the user requirements. We generally know how to define the main functions for the normal use of the product or a system, but it is extremely difficult to express the functions that are linked to the context of the use of the product, because they are often implicit functions (Tichkiewitch and Roucoules, 1999). Generally speaking, it is easier for a customer to express what he/she does not want, but it is difficult to define what is expected.
Requirements can be classified into several categories, as follows (based on Grabovski et al (1999)):

- Depending on their origin, it can be differentiated between external requirements (given by customers) and internal ones (found during the design process).
- A distinction can be made between explicit and implicit requirements. Explicit requirements are given directly at the beginning of the design and must respond to the question 'what should be achieved or avoided ?'. Implicit requirements are derived from these and usually surface during the clarification of the design.
- Complex- and elementary requirements may also be distinguished, where elementary requirements can be derived from complex ones by subdivision.
- Fixed requirements must be strictly satisfied ('need to have') whereas desired requirements can be described as 'nice to have' features of a product or a system. If fixed requirements cannot be met, they must be either explicitly changed or rejected.
- A further distinction can be made between quantitative and qualitative requirements. Quantitative requirements are more operational due to their unambiguous and precise formulation, and can be measured or described by mathematical expressions (criteria). In contrast , qualitative requirements can only be evaluated indirectly.

A clarification of the requirements is usually necessary *before* a preliminary design can take place. However, further clarification activities may be involved - e.g. whenever design is stopped due to the discovery of imprecise and/or infeasible requirements. Note that if the designer(s) follow a set of design principles, then these principles themselves *generate* questions that users need to answer, i.e. the design team may help users state their relevant requirements before a preliminary design.
Sometimes at the start of design, user requirements not only contain functional requirements (such as required functionality or capability,...) but also desired solution elements (existing resources such as machines, information technology components,...). Design in this case is not only a process of mapping requirements onto a possible solution, but also a process of refinement - simultaneously completing both the requirements and the solution.

15.2.2 Migration from AS-IS to TO-BE

Design consists in specifying the future system called TO-BE. The TO-BE is usually expressed in the same semantics as the requirements and the AS-IS (i.e. AS-IS and TO-BE modelling is using same language and formalism). Preliminary design tends to transform requirements to an ideal system specification without taking into account various constraints, such as costs or feasibility. Generally speaking, the TO-BE system is different from the ideal system envisioned by the user, because before completion, one must migrate AS-IS towards TO-BE; this presents risks, generates costs and may cause delay. The design project is then limited by constraints and resources of the user. Thus, in addition to the migration from the AS-IS- to the TO-BE state, there is also a migration from the idealised future system- to the TO-BE state, as illustrated in Fig. 15.6.

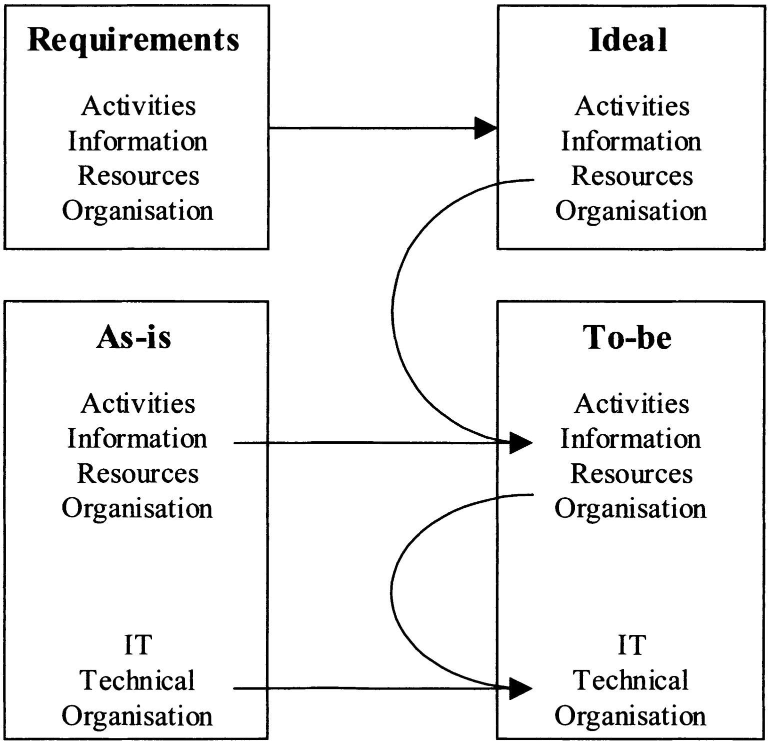


Fig. 15.6. Migration from ideal to TO-BE and from AS-IS to TO-BE

15.2.3 Main Issues Taken Into Account During Preliminary Design

Designing manufacturing systems is not only a technical issue. Various factors and constraints may influence design decisions, such as financial implication, risk, compliance to standards, external restrictions, flexibility and feasibility etc.

15.2.3.1 Economic Estimation

An economic analysis allows to perform a rough-cut calculation to estimate costs and benefits on solution principles adopted. According to Williams et al (1996), this economic evaluation must be tightly linked to future enterprise business objectives and plans, as well as possible long term development strategy. Costs should be compared with potential benefits to be generated by implementing the design solution. Benefits can be *tangible* or *intangible*. Tangible benefits usually consist of production volume, labour costs and raw material costs. They are traceable by the accounting system. Intangible benefits, such as quality, market share and customer satisfaction are often ignored by the economic evaluation. The companies' lack of interest in these issues stems from the lack of accounting tools for the evaluation of complex projects. Nevertheless, these issues must be taken into account as well.
Some of the known economic evaluation techniques are :

- Payback period: This is the length of time necessary to recoup initial investment. The shorter the payback period, the lower the risk;
- Return on Investment: It shows how effective a company is in generating income, given the amount of investment. It is calculated as Net income/Total investment, e.g. the higher the percentage, the more favourable the project;
- Internal Rate of Return: This Return is the interest rate received for an investment consisting of payments and receipts occurring at regular intervals. In other words, it is the percentage rate that makes the present value of costs equal to the present value of benefits.
- Net Present Value: It is based on the idea that an amount of cash is to be paid (or received) in the future has less value today than when it is actually paid (or received).
- Discounted Cash Flow: The sum of the values of current cash-flow associated to a project of investment. It allows to measure the enrichment of the enterprise with respect to the realisation of a project.

More details about these economic evaluation techniques can be found in (Brealey and Myers, 2000).

15.2.3.2 Risk

Risks must be evaluated in the preliminary design phase when basic solution principles are determined. Generally speaking, risks are broader than only financial risks (Williams et al, 1996). For example:

- Technology-related risks : Some advanced IT or Manufacturing technology may not be at a satisfactory degree of maturity. Implementing such technologies may imply frequent updating (e.g. software), dis-functioning and high cost of maintenance and repairing.
- Organisation-related risks: Sometimes, implementing new technology requires to reconsider the organisation and its working procedures. For example, by introducing computer-based planning and manufacturing systems, the organisation may need to change from an old hierarchical management structure to a more autonomous, decentralised organisation.
- Human-related risks : Implementing new system might encounter resistance to change coming from personnel. Sometimes, existing employee qualification does not meet the required standards to run the new system, and therefore the potential of these new technologies can not be promptly exploited when the system is implemented.
- The preference of the top management for some particular technologies or vendors may also influence the choice of a particular design solution. Bad growth forecast and unrealistic business objectives, leading to errant design requirements are also sources of risk for design.

15.2.3.3 Standards

ISO and IEC (International Electro-technical Committee) are main sources of standards. Some industrial bodies such as ISA (Instrument Society of America) and IEEE (Institute of Electrical and Electronics Engineers) are the sources of industrial "consensus". Non-profit organisations such as OMG (Object Management Group) and OAG (Open Applications Group) are also proposing IT standards. Generally speaking, standards should be used at two levels:

- System engineering and integration standards that are used to carry out a design project such as ISO 15704: Requirements for Enterprise Reference Architectures and Methodologies and CEN 40003.

- Solution standards that can be used to specify both IT and Manufacturing Technology solutions such as STEP for product specifications, MAP/TOP for network interconnection, MANDATE for manufacturing data exchange...

The use of standards allows to promote portable components and interfaces, and favour interoperability. The use of standards also eases system implementation and maintenance by using Plug-&-Play components.

15.2.3.4 External Restrictions

External restrictions are constraints originating from outside of the realm of stakeholders in the system under consideration. Some examples of these restrictions relevant to manufacturing system design are as follows:

- Regular law and government legislation such as working hours, dangerous products manipulation, gas emission directives, safety, human relations etc.;
- Local culture / way of life and education level;
- Compliance with standards of good community behaviour (good neighbour, citizen of the world,...);
- Compatibility and interoperability with various partners, including maintaining good relationship with suppliers and customers, even competitors;
- Environmental constraints such as climate (floods, high temperature,...), earthquake and other dangers of natural phenomena, etc.

These factors are as important as technical and technological considerations. Many past experiences have proved that ignoring such factors may lead to 'perfect technological designs' but unfeasible implementations.

15.2.3.5 Flexibility

Flexibility is related to the notion of re-configurability of a system in order to manufacture new products. The desire to design a flexible system should not be considered as a goal in itself but as means to satisfy present and anticipated future business objectives.

Generally, when a system is flexible, it is less specialised and therefore may be less efficient. Customer functional requirements are the main source to determine the degree of desired flexibility. For a given system, one must determine the limits of the degree of flexibility and efficiency, and this can be contrasted with the degree of efficiency that current technology can offer without the constraint of flexibility (usually resulting a dedicated but not flexible system). The chosen degree of flexibility is evident and must be determined by requiring a minimum efficiency. In general, a number of criteria must be taken into consideration to determine the planned degree of flexibility, such as cost, risk, maintenance, personnel qualification and so on.

15.2.3.6 Feasibility

According to the PERA approach (Williams, 1994), the feasibility study views the proposed solution from several aspects: (1) have the integration goals been met? (2) have the timing requirements been achieved? (3) have the necessary levels of accuracy been obtained? (4) Is the necessary technology available? (5) Can the system be implemented in a cost-effective manner? The feasibility study should involve all stakeholders (e.g. in the case of a manufacturing system design / implementation project: the design team, the project's steering committee and the end user).

Feasibility checking can also be extended to the following domains:

- Easy to maintain. Constraints and requirements for system maintenance;
- Ergonomic considerations including the working condition of operators;
- Recycling requirements;
- Sensitivity analysis (how much any anticipated variables will effect the operation and operational characteristics of the system, such as changes in demand, personnel, costs, environmental variables).

15.3 From Function & Data to Resource & Organisation: The Assignment Issue

After the required functionality of the future system has been defined, one of the most important design activity is to map functions onto resources capable of performing the functions. Suppose that the main functionality of the system is concerned with manufacturing. An organisational structure must also be established to define various responsibilities with respect to functions and resources.

15.3.1 Assignment of Functions to Organisational Entities (Services)

The problem here is to allocate responsibilities for all necessary functions. An organisational entity being responsible for a function means that the organisational entity has authority over all resources necessary for the given function. This authority gives the entity some level of autonomy. For example, if a function FA is within the authority of an entity, then the entity should be able to control a resource RA so as to carry out that function FA. The entity should also have a resource RC that can provide the required control function FC.
Note that theoretically, an organisational entity is a system (or subsystem) and should can be considered as a heterogeneous and autonomous resource

itself. It is clear that a machine tool is a resource, while a factory is an organisational entity (because it is an autonomous resource). Where we draw the line between 'organisational entity' and 'resource' depends on the structure of the system in question. As long as a system is an autonomous system, it can be considered an organisational entity. Organisational entities may consist of lower level entities, which does not contain any more autonomous entities, only a mixture of human and automated resources. Whereupon organisational entities have responsibilities due to their autonomy and authority, ordinary resources only have functions, but no responsibility.

From the above exposition it is evident that after allocating functions to resources (discussed in the next section) and consideration regarding responsibility allocation, which functions will be implemented by humans, (as lowest level organisational entities) not only have functions but also autonomy and authority - thus responsibility.

Since an organisational entity in general includes both humans and automated resources the issue is not the same as the allocation of functions to resources (discussed in the next section) albeit related.

One organisational entity may be responsible for several functions. The assignment consists of specifying which organisational entity is responsible for what function and associated data. For example, who is responsible for a function top management (centralisation), an external provider (outsourcing), or a site of the company (decentralisation)?

An assignment matrix may be used to assign responsibilities, such as shown in Table 15.1:

Table 15.1. Assignment matrix

	Org entity 1	Org. entity 2	Org. entity 3	Org. entity 4
Function 1	X			
Function 2		X	X	
Function 3	X			
Function 4				X

The case demonstrated by 'Function 2' sometimes arises, where two 'organisational entities' need to provide the same function. This results in a shared responsibility, raising questions of accountability. Therefore it is better practice to decompose/detail the function in order to precisely specify ownership of the responsibility of entity 2 and entity 3. This decomposition is not necessarily function based, e.g. the same function may be provided by two different organisational entities, but the functions are performed on two *disjoint* sets

of inputs (e.g. organisational entity 1 provides maintenance function for site A, and organisational entity 2 provides the same function for site B).
After functional decomposition, another preliminary design activity is to specify distributed sub-systems / functions. This is illustrated in Table 15.1. Functions F1.2, F1.1.1 and F1.1.2 are regrouped, they are to be implemented at site A. Function 1.3 will be implemented at site B. In some cases, a detailed function decomposition will be needed at this stage, e.g. if part of a function is to be allocated to site A, and another part to site B.

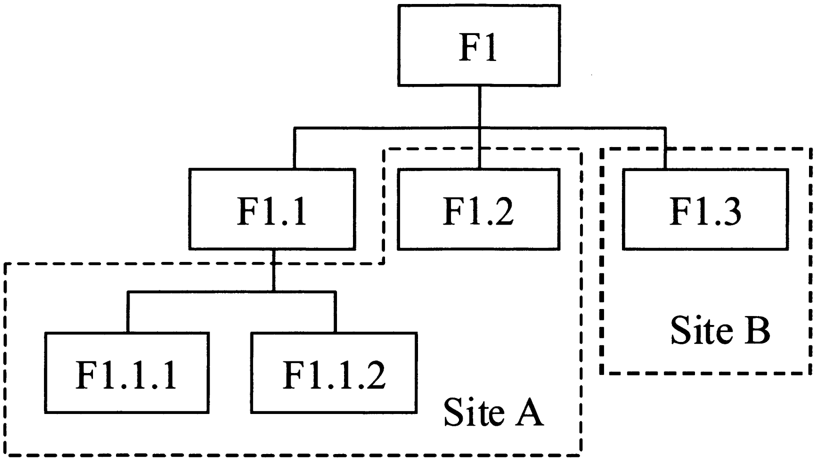


Fig. 15.7. Distributed sub-system / function specifications

Like functional distribution, data also needs to be distributed where needed, in order to provide the necessary information. For example, if the modelling formalism used to represent information requirements is the entity relationship (ER) model, data entities are to be grouped and arranged according to the location of use of these data entities. External schemata may be defined as views of the overall information schema for each use, i.e. an external schema may be developed for each location (what data must be available at which site). These external schemata can then be used for defining data distribution. However, further detailing of external schemata is necessary to define data access requirements for each major application.
In order for data distribution to be determined (including decisions on the replication of certain data) it would be necessary to have specific information about the volumes of data access transactions, as well as essential response times. The dilemma is that to obtain this information it would be necessary to go into detailed design, and thus unduly delay the completion of architectural design. Therefore architectural design should be based on a data management principle which allows data distribution to be *independently* defined

from other design decisions, and relegate initial data distribution decisions to detailed design as well as allowing data distribution decisions to be made during operation (such as through database tuning).
The distributed implementation of functions leads to a distributed information system. When deciding on such a design solution, several factors should be taken into account:

- Distributed information storage still requires that each data element should have only one 'original' instance. End users, for reasons of convenience and functionality (such as minimum required access speed), often need to duplicate data and this can potentially lead to various anomalies. Every data element must have one 'original' copy and other storage of the same data element must be under a controlled replication process that ensures required data consistency. Architectural design must ensure that the selected underlying data management solution is capable of distributed data management, including the control of consistency of replicated data.
- Distributed systems may or may not generate more significant volumes of information exchange than centralised system do. In the optimal case, data distribution increases the availability of data. However, communication costs and delays may result from the same data being necessary at multiple sites, where this need heavily depends on the characteristics of the system and processes in question. Thus architectural design needs to consider associated costs, such as the cost of network, maintenance etc.);
- Information exchange between sites may introduce delays into data access. This delay may become non-negligible for some types of industry. Some applications may need to access data from multiple sites, such as decision support systems / data warehouses that aggregate data from across multiple locations. However, these systems often do not require fully up-to-date information, and can rely on snapshots of operational data taken at regular intervals and asynchronously forwarded / distributed to sites where they may be needed, thus alleviating the need for high speed high bandwidth connections. Therefore, architectural design usually considers separating operational data management from the management of decision support data.

15.3.2 Determining the Level of Automation

The functions previously identified may be performed by humans or by automated means (computers and machinery[9]). One of the design options in architectural design is to determine the level of automation. For a given function, the designer must compare the actual technology available with the human abilities in terms of speed of response, physical strength, working conditions, cost etc.

[9] in the following, the terms computer, machine, automated machine will be used interchangeably with the meaning of non-human resource.

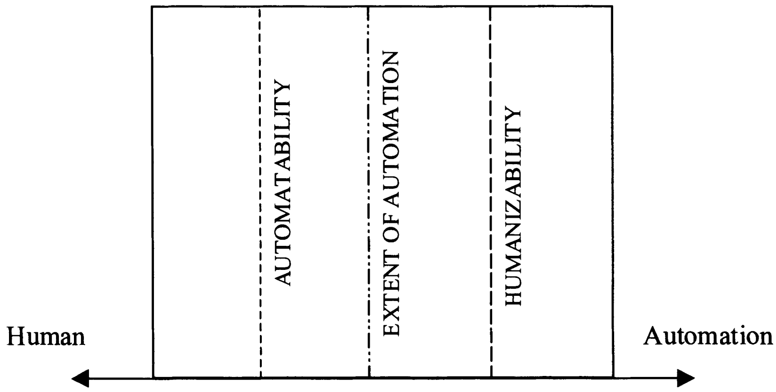


Fig. 15.8. Determining the level of automation (adapted from Williams (1994))

Fig. 15.8 shows intuitively how to determine the level of automation. For a set of functions, the *'Automatability'* line represents the extent to which automated technology may be used. Automatability is limited by the fact that many functions require human innovation, or other abilities that cannot be automated with present available technology (Williams, 1994). As opposed to this, the '*Humanizability*'line shows the limit to which humans can be used to perform the set of functions. Beyond this limit, machines must be used to replace the human. The actual design decision is then schematically represented by the '*Extent of automation*'line, which is a balanced solution between the limits of automation and the limits of human capability.
To determine the extent of automation, a number of criteria should be considered, such as:

- Cost of using automated technology vs. humans,;
- Maturity and state of evolution of technology (especially for computer software and hardware), and associated risks (to push or not to push the 'design envelope');
- Human working conditions (dangerous product/environment, existence of legislation preventing human from working in certain conditions, union legislation);
- Regularity, reliability and flexibility considerations (human is more flexible, while automated machine provides regularity and quality; humans are available during set working hours while machines may be available at any time, etc);
- Workforce characteristics, local economy, regional politics (availability of trained /affordable workforce, existing need to provide jobs to support local economy, etc).

15.3.3 Assignment of Functions to Resources

According to the chosen position of the 'extent of automation' line, a manufacturing function may be assigned to either a human resource or a machine resource.
One of the problems in assigning resources to functions is the difficulty to precisely define the required manufacturing resource capacities, because the required manufacturing functions are usually defined according to the set of products to be produced by the enterprise. However, in the current economic and market environment products constantly evolve, and as a consequence the manufacturing processes and functions also evolve.

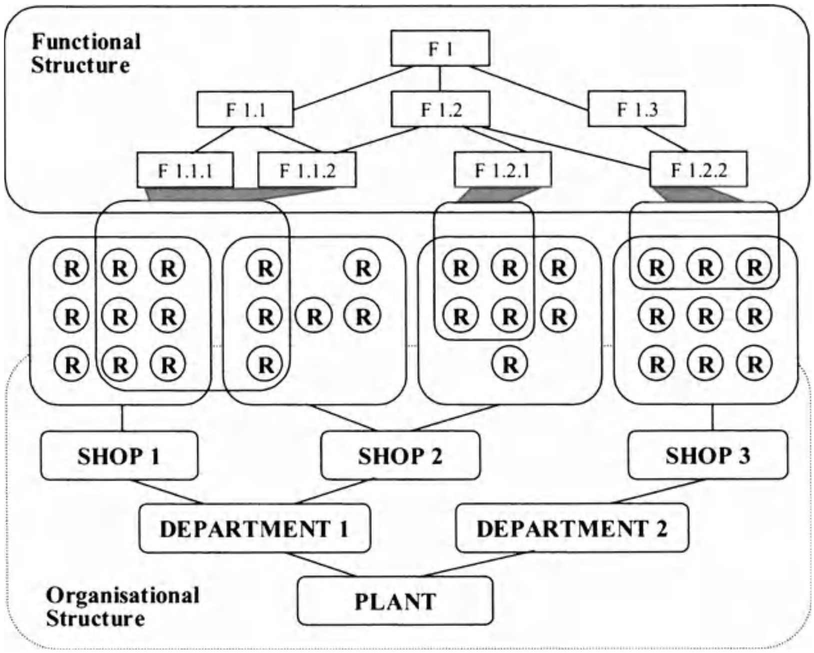


Fig. 15.9. Relation between resources, functions and organisation (adapted from
(AMICE , 1993)

Information Technology-related resources to be specified are of three types: (1) data storage devices (including hard disk, CD and optical recording facilities); (2) data processing devices (PC, mainframe,...); (3) communication and data transmission devices (for example, Local Area Network (LAN),...) and (4) presentation devices (screens, printers, projectors). As indicated before, at this preliminary design stage, only solution principles need to be determined.

For example, one can use a network of decentralised PCs interconnected via a LAN instead of a mainframe with distributed terminals.
The type of data storage devices also needs to be determined without choosing a specific, commercially available system. For example, it could be manual or computerised. For computerised one, it can be the centralised or distributed implementation. The database itself can be a relational type or object-oriented, etc.
The assignment of resources to functions is one of the most important issues of the preliminary design because resources are the main components of the system structure. This structure will determine the main functionality and behaviour of the designed manufacturing system. For an optimal assignment of resources to functions several remarks and recommendations must be considered:

1. When several functions are assigned to one resource, it is necessary to regroup them. This can be done using the following rule: if two functions are to be executed continuously in time by the same resource (no-interruption), one can regroup them into one function.
   Generally speaking, when mapping functions onto technology or human implemented sub-systems, it may be necessary to rearrange the functional decomposition structure. For example, some functions of the same nature may need to be aggregated into one sub-system to be automated; some others may be aggregated into another sub-system, e.g. to be implemented by humans.
   Aggregating functions and data into human- and technology implemented parts is an important design activity, also called *structural* design. One of the output of the structural design is specifications of the physical system. Requirements on physical manufacturing operations (for example, turning, drilling, milling etc.) will be transformed into either machine- or human resource specifications. This is concerned with the type of machine (NC or conventional), the number of machines needed and technical capabilities required, without choosing specific commercially available products. The final choice will be made at the detail design stage.
2. To specify the physical manufacturing system, some known techniques / models can be used to analyse the possible solutions (e.g. the Production Flow Analysis technique, Group technology etc). Figure 15.9shows the relationship between functions, resources and organisation structure.
   Machine and human resources are identified according to required capabilities to execute functions. Resources are then grouped into cells, for example according to their nature (machining cell, thermal treatment cell,...). Cells can be grouped into various shops which in turn constitute the plant. At this point it is therefore possible to finalise the allocation of responsibilities and authority as noted in Section **15.3.1**. For example, the responsibility and authority to run and to maintain cells and shops is defined and the organisational structure / organisation chart is completed.

3. Two functions with different outputs can be technologically similar. In this case, one can define a generic function that can be parametrically controlled to produce either of the desired outputs (also called 'parameter-transformation' - refer Fig. 15.10).

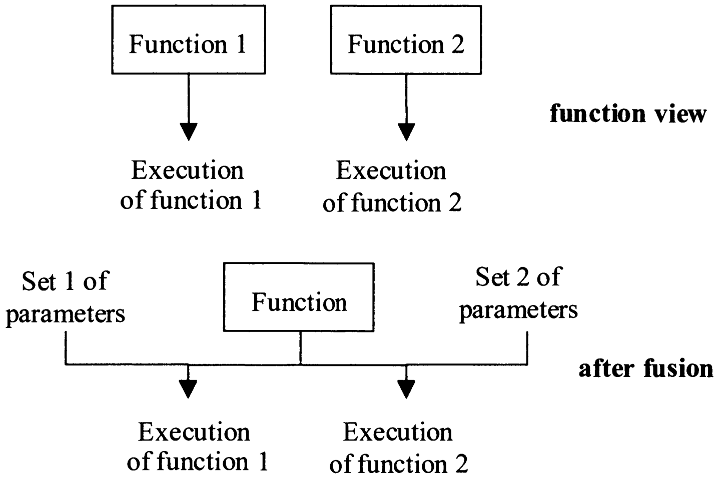


Fig. 15.10. Aggregating two similar functions using parameter-transformation

4. The work capacity of a team is not proportional to the number of its members: there is a loss because of necessary additional co-ordination activities (refer Fig. 15.11). It is necessary to consider this overhead when assigning a function to a team composed of human resources.
5. The assignment of a decision to a decision-maker must take into account the quantity of information the decider must manipulate in order to make the decision (cognitive limitation). If there is too much information, data aggregation is required. Data aggregation is an important issue of preliminary design. For example when designing a production planning / control system, shop-floor raw data are collected at a level of detail needed for humans operating at that level. This data must be aggregated for managers at the higher level. e.g. a Master Production Schedule contains less detail than the sum of schedules for each machine and operator, but its structure and content is specific and purposeful for decision-making at that level.
   Another design option concerning data is to decide the portion that will be stored in databases and the section of data that will be collected and stored on paper. This design decision should be made simultaneously on the automation of the functions which produce and use the data.

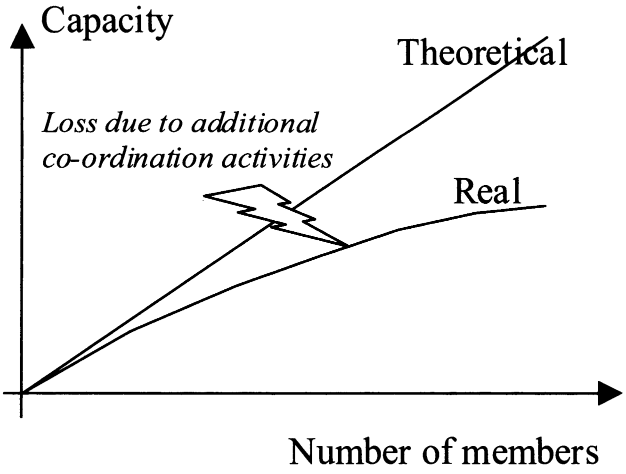


Fig. 15.11. Capacity lost due to co-ordination activity.

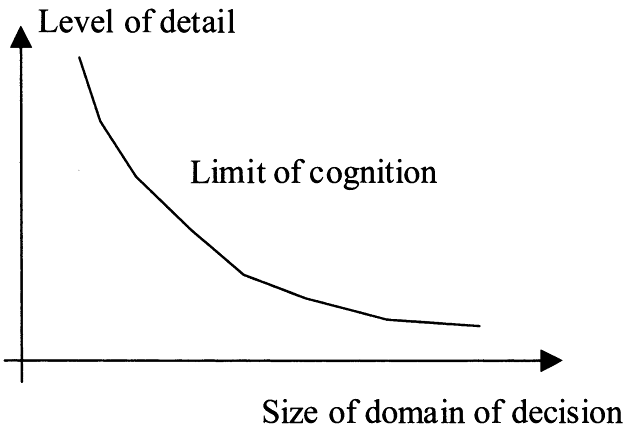


Fig. 15.12. Cognitive limitation of information processing.

6. Economic evaluation of the function and comparison of cost/advantage. For each design decision about aggregating two or more functions into one generic function, it is necessary to evaluate the two possible solutions according to what advantage or disadvantage this aggregation has on the system structure, cost, risk, ability to use standards, maintainability, external restrictions, flexibility, feasibility etc.
   For example, for cost evaluation, a diagram of cost indicators can be constructed as shown in Fig. 15.13:

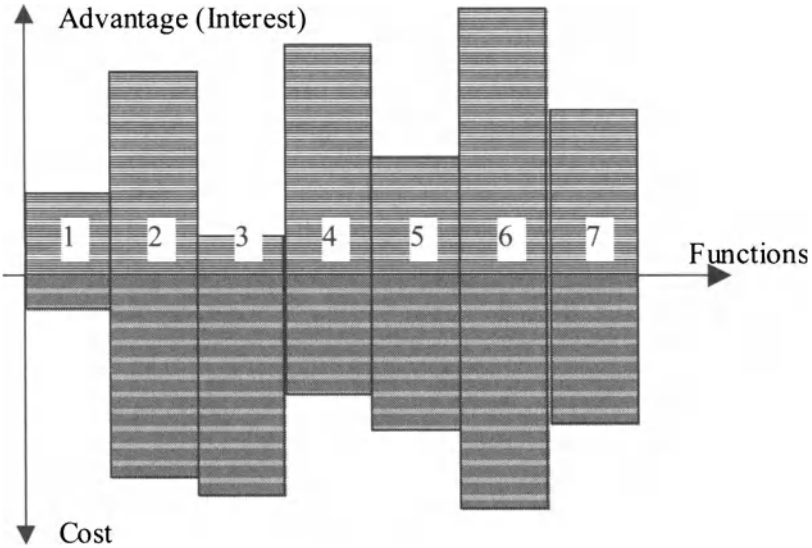


Fig. 15.13. Cost evaluation and comparison.

In this figure it can be seen that function 6 costs a lot but will be a big interest for the system. Function 3 is also expensive, but it seems to have a very little interest. It is therefore important to precisely evaluate if it is necessary to keep this function. The interest of the function 1 is quite small but its cost is also very small. It is thus acceptable to keep this function.

15.3.4 Design Information Technology Infrastructure

Before going into detail of infrastructure design it must be pointed out that infrastructure design is a separate systems engineering activity, and therefore the same principles apply as demonstrated in Sections **15.3.1-15.3.3**. When responsibilities and resources are allocated to functions, in fact only the 'immediate' functions necessary to implement the desired business processes are considered. However, all resources depend on an infrastructure to be able to operate. For example a computer-aided design tool relies on a computer operating system (in general a software infrastructure, or common operating environment), the common operating environment also relies on a hardware infrastructure, which in turns depends on a building infrastructure (buildings, roads, heating/cooling, electricity), etc.
The hallmark of an 'infrastructure' as opposed to a mere subsystem is that infrastructure is ubiquitous and transparent to the system using it. Infrastructure functions are (within limits) available anywhere needed in the system; should the system be reconfigured, the infrastructure may remain unchanged

or can be independently upgraded without too much effect on the design decisions during system restructuring. Infrastructures are complex systems in themselves. When an infrastructure is designed, the same principles apply as to any other system; therefore e.g. a mission and concept must be developed. Infrastructure requirements need to be collected and analysed, architectural design of the infrastructure must be carried out and finally detailed design and implementation must follow. Thus, there is a *recursion* of systems, well known to systems engineers.
This Chapter includes two sections specifically addressing architectural design of infrastructures, mainly because the requirement to be transparent and ubiquitous is a special characteristic of infrastructures, and therefore specific *additional* principles apply over and above the principles discussed in Sections **15.3.1-15.3.3**.

15.3.4.1 Design Information Technology Infrastructure

With increasing demand on software portability and inter-operability of enterprise processes and activities, the design of an Information Technology Infrastructure becomes an important issue. This infrastructure does not provide customer services but allows distributed functions, data, resources and organisation to work as an integrated whole. The discussion is restricted to the example of Integrating Infrastructures necessary for manufacturing and business process integration in general, without covering basic computer software (such as 'Common Operating Environments' (COEs)), hardware or communication infrastructure.
Since integrating infrastructures need to be 'infrastructure-like', there is a long list of constraints that such a system must comply with. The designer of the integrating infrastructure must ensure the compatibility with a large number of standards, or otherwise the design will not be usable in a wide range of applications; thus, it will not be transparent, and therefore it must be purpose-built as a subsystem, hence losing one of the main characteristic properties of an infrastructure! As a result, integrating infrastructures must be based on commonly accepted reference models, or standards.
As of today, the functions that an integrating infrastructure must provide are not part of common operating environments (basic operating system and surrounding services), however, it can be predicted that there will be a trend to incorporate all these functions in future COEs.
The international standard ISO 15704 - Requirements for Enterprise Reference Architecture and Methodologies presents a reference model based on a European pre-standard ENV13550. This model expresses the capabilities of environments for developing, executing and integrating enterprise models on an open IT based platform. The services needed to provide these IT services are referred to as EMEIS (Enterprise Model Execution and Integration Services) as shown in Fig. 15.12. A standard for EMEIS is being elaborated at a

high level of abstraction (functionally) and does not specify how services are to be implemented.

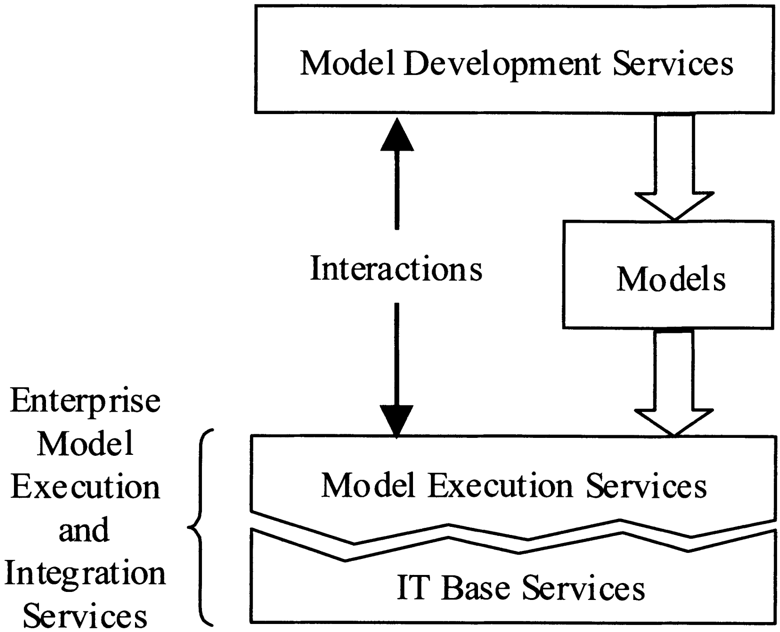


Fig. 15.14. The Reference Model for EMEIS.

The Reference Model consists of three layers:

- the Model Development Services (MDS) which is the environment to build and test enterprise models before releasing them for execution;
- the Model Execution Services (MES) which transform the enterprise model into executable software and provide all necessary CIM specific services so that the executing model can control enterprise resources;
- the IT Base Services which are not CIM specific but are concerned with basic inter-working and interconnecting services (the 'Common Operating Environment').

One of the issues is to design interfaces between various components of the system. Components are of three types :

- Human type (operator and manager);
- Machine type (including sensors, automated storage and transport subsystems and modules);
- Computer type (including applications and databases).

Consequently, the interfaces are interconnections between these resources. The interconnections are electronic and/or mechanical, which permit two or more physical (or human and organisational, or both) modules to carry out the information and material and energy transfer functions of the two or more functional modules which are interconnected.
In Hong and Williams (1994), some important recommendations for designing interfaces are formulated as follows (with modification):

- The method of processing information requests and then transmitting the result back to a user should be transparent to the users;
- All users should have the same method of accessing information, regardless of where they are connected to the network;
- Interfaces to the outside world (customers, suppliers, government agencies, capital providers, etc.) should be standardised;
- Interfaces, protocols and interactions across all levels of management, support and organisational systems should be standardised for control, data storage, retrieval, and communications;
- All major functions should have a standard format for I/O which could be used by any vendor supplying software to perform that function;
- Enterprise integration system communications should follow OSI, MAP, Field Bus or other established ISO and IEC standards in order to promote the open system interconnect capability.

15.4 Conclusion

The preliminary design translates requirements in solution principles that will be further described at the detail design phase. The main issue of the preliminary design is to determine what types of resources to use so that the future system produces the functionality and behaviour required by users. Awareness of the solution principles and resource types, as well as the refinement of requirements are crucial to obtain a successful design. It is not only a technical issue, but also involves negotiation, teamwork, project management, extraction of the knowledge of factory personnel and taking into account various factors such as economic, ecological, social etc.

References

AMICE (1993). CIMOSA: Open System Architecture for CIM. Springer-Verlag, Berlin, 1993.

Brealey, R.A., Myers, S.C., "Principles of corporate finance", Mac Graw Hill International Editions, December 2000.

Chen, D., Vallespir, B. and Doumeingts, G. - "Enterprise modelling and engineering : some complementary requirements on standardisation", In CEN

TC310 WG1 workshop on Enterprise evolution and engineering, May 24-26, 2000, Berlin.

Doumeingts, G., Vallespir, B., Zanettin and M., Chen, D. (1992). GIM: GRAI Integrated Methodology for designing CIM systems. GRAI/LAP, University Bordeaux I, version 1.0, May.

Doumeingts, G., Vallespir, B. and Chen, D. (1998). Decision modelling GRAI grid. Chapter in : Handbook on architecture for Information Systems (Peter Bernus, Kai Mertins, Gunter Schmidt. (Ed)), Springer Verlag.

Eynard, B. (1999), Modélisation du produit et des activités de conception : Contribution à la conduite et à la traçabilité du processus d'ingénierie. PhD dissertation, University Bordeaux I, June 30, 1999.

Grabowski, H., Lossack, R. and El-Mejjbri, E. (1999), Towards A Universal Design Theory, Proceedings of the CIRP International Design Seminar, University of Twente, The Netherlands, March 24-26, 1999.

Hong, L., and Williams, T.J. (1994), A formalisation and extension of the Purdue Enterprise Reference Architecture and the Purdue Methodology, Report number 158, Purdue Laboratory for Applied Industrial Control, December 1994.

IFAC-IFIP Task Force (1999). GERAM: Generalized Enterprise Reference Architecture and Methodology. Version 1.6.3 . http://www.cit.gu.edu.au/~bernus (also as Annex to ISO IS15704 :2000)

IS 15704 (2000). Requirements for Enterprise Reference Architecture and Methodologies.

Pahl, G., and Beltz, W., (1996), Engineering Design - A systematic Approach, Second edition, Edited by Ken Vallace, Springer-Verlag London limited, 1996.

Tichkiewitch, S. and Roucoules, L. Methodology for innovative design, CIRP International Design Seminar, University of Twente, The Netherlands, 24-26 March 1999.

Vernadat, F. B. (1996). Enterprise Modelling and Integration – Principles and Applications. Chapman & Hall, ISBN 0-412-60550-3.

Williams, T.J. (1994). The Purdue Enterprise Reference Architecture, In: Computer In Industry, vol. 24. 1994.

Williams, T.J., Rathwell, G.A. and Li, H. (1996), A Handbook on Master Planning And Implementation for Enterprise Integration Programs, Report Number 160, Purdue Laboratory for Applied Industrial Control, June 1996.

16

ORGANISATIONAL DESIGN

Peter Bernus

Griffith University (Australia)

16.1 Introduction

This chapter is an introduction to organisational design; more authoritative texts exist that expand on the subject in detail. In this chapter, organisational design is treated as part of the enterprise architecting process. Organisational units are abstract entities with their own identity (some also being legal entities), and consist of people, resources and a set of responsibilities and allocated authority. Organisational units are created for two reasons: to shield the complexity of human relations and responsibilities that inevitably arise within a company, and to be able to temporarily disassociate oneself from the identity of individuals who are part of the organisational unit. This gives the organisational unit a certain level of stability – e.g. it is possible to refer to a given role in an organisational unit, instead of referring to the particular person filling in that role at the time. Some organisational units are created to exist for a longer period of time, such as a department or a committee, while some may be created for shorter periods, such as task forces, ad-hoc committees, project teams, etc.
The basic question in organisational design is:

- How to aggregate management and production roles into organisational units?
- How to define the relationships between these roles?
- How to allocate individuals to roles in the unit?

Organisational design literature has for long suggested (Thompson , 1967; Tushman and Nadler , 1978) to disassociate these concerns – implying (implicitly) that the decision to form organisational units should be purely based on the function of the organisational units (decisional, production and service tasks). Once practicing managers are asked, however, whether this is a good idea, it immediately transpires that designing the organisation with a disregard to available human resource and their individual characteristics is likely to lead to inferior or even unworkable designs.

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

This chapter advocates a method for organisational design, developed by Goold and Campbell (2002), that gives equal concern to the functional needs of the company (i.e. the tasks that must be performed) and the human resource possibilities (available or likely to be available individuals). The outline of this method is given in Sections 16.3.1, 16.3.2 and 16.3.3.
The concern with the personal characteristics, aspirations, relationships, motivations and skills of existing personnel leads to another important question. If organisations must continuously and dynamically change, then who should have the authority and responsibility for making changes and what competencies are needed from the manager who does change the organisation? In any reasonably large organisation, the task of organisational design cannot be performed by a single top manager or management team. There are a number of reasons for this:

- Such an approach would take away authority and autonomy from lower level management;
- Top level management does not possess sufficient information to make decisions on the detailed organisational design level, and
- Due to the complexity of the task, there is not enough time to do so.

Thus, organisational design is a *distributed* task, where top level management's influence must be exercised though company-wide principles and rules, rather than direct intervention into the details of organisational design on lower levels; only a limited level of design should be done on the corporate level. There is little doubt that top management must develop an organisational design that passes the test of fitness for the purpose (and demonstrate the adherence to some good design principles). However, organisational design must be further detailed and refined by management lower in the hierarchy. As a consequence, organisational design skills and authorities need to be clearly understood. It must be ensured that those lower level managers who need to work out further details of the organisation (and continuously change it) have adequate design skills as well as the motivation to do so.

16.2 The Relationship Between the Decisional and Organisational Structures

16.2.1 Mapping the Decision Structure to Organisational Structure

Chapter 12 has treated the functional / decisional view of the organisation, defining the tasks of management, but temporarily disregarding how these tasks should- or can be aggregated into an organisational design. Chapter 15 (about architectural design) has discussed the generic principles of aggregating functions (production, service or management), and pointed out that there are many ways to achieve this aggregation. When considering the aggregation,

for example, of manufacturing functions into manufacturing equipment, the choices are made according to the characteristics of possible machinery (such as cost or purchasing and running, flexibility, quality, etc). Furthermore, the way this aggregation is done, the resulting system will have systemic properties that *emerge* as a result of the way the aggregation was done. These systemic properties are important, because they are closely linked with the strategic intent of the company; e.g. uniformity in production equipment (due to aggregation of production functions into general purpose equipment) creates the possibility of centralising most maintenance and service tasks into a common part of the organisation.

In organisational design, one has to aggregate human functions: (a) into big aggregates (organisational units) and (b) into small aggregates that correspond to a list of responsibilities that can be allocated to individuals in the company. In doing so, choices are made according to the characteristics of people (their skills, motivations, relationships, etc) and the emerging design will carry strategically important systemic properties of the organisation.

In fact, organisational design has already started in the identification phase of the enterprise (refer to Chapters 4 and 7). This is because the management's strategy has identified the main building blocks of the Business Model, calling these 'enterprise entities', at the time when the strategic relationships among these entities were determined.

Each decision framework is a link between people and represents what *kind* of objectives one should be able to determine for the other, under what kind of constraints, and what kind of decision variables are made available for the subordinate to manipulate to achieve the objective. The actual 'values' of these 'kinds' are determined during the operation of the organisation. For example, a management role $\mathbf{B}_0$ that has the task of deciding product strategy in the organisation may provide a decision framework for $\mathbf{B}_1$ that has the task of deciding on the weekly production tasks as well as the operational control of product delivery.

Figure 16.1 shows a decisional structure (refer to Chapter 12) and how decisional tasks are allocated to organisational entities (i.e. an individual or a group of individuals).

The corresponding organisational structure (with jobs numbered A_0, A_1, A_2, A_{21}, B_0, B_1) is shown in Fig. 16.2. The management tasks (shown as rectangles) represent the duties/ responsibilities of the person, and are presented in the form of a job description. The solid lines of the chart show decision frameworks that – as a result of this particular mapping – become supervisor-subordinate relationships, while the dotted lines represent decision frameworks that are internal to the person's responsibilities. Each such decision framework is a link between people[1] and represent what *kind* of objectives one should be

[1] In case of a dotted arrow the objectives, decision variables and constraints describe a decision framework that a person with *integrity* makes, and later respects in carrying out a lower level duty. Such 'internal' decision frameworks may have

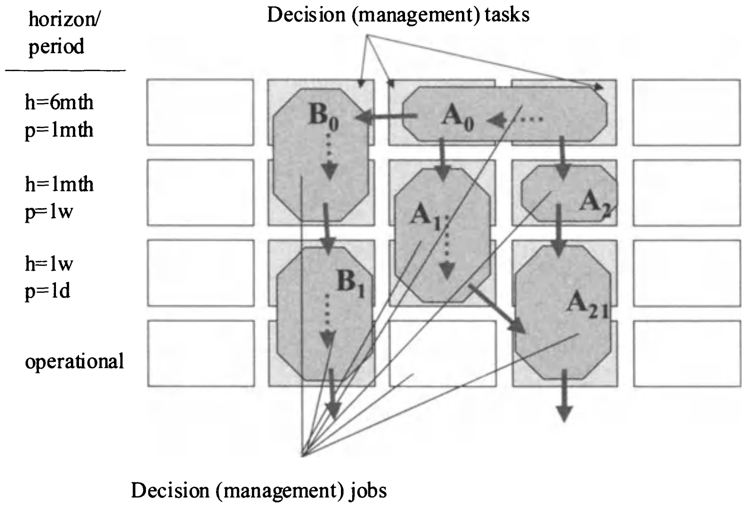


Fig. 16.1. A decisional structure with tasks allocated to organisational entities

able to determine for the other, under what *kind* of constraints and what *kind* of decision variables are made available for the subordinate to manipulate to achieve the objective.

The values of these kinds are determined during the operation of the organisation. For example, B_0, which has the task of deciding product strategy in the organisation, may provide a decision framework for B_1, that asks B_1 to decide on the weekly production tasks, as well as the operational control of product delivery. The kind of objectives that B_1 must be prepared to handle may be to '*to ensure that customers get timely confirmation of their orders*' and '*to ensure that orders from suppliers are sent out in time in accordance with the production schedule*' (etc). The actual value of these kinds of objectives may be '*to ensure that the London customers get priority next week*' (because the production plan is tight and the London costumers are very important). The kind of decision variables and constraints may be '*in carrying out the weekly production- and order scheduling B_1 is allowed to select any of the suppliers that have an approved supplier status, or, if the demand can not be met using preferred suppliers, then up to a given percentage of incoming goods may be ordered from suppliers without such status*'. The actual value of this decision variable is '*the list of currently approved suppliers*' and, from time to time, B_0

to be referred to in the negotiation between two individuals and should be 'externalisable'. This is also important in case the given job is re-distributed and (part of) the tasks allocated to one person may potentially become the duty of another person.

may determine the '*actual percentage of suppliers who are not on the preferred supplier list*'(thereby determining the level of risk the company is prepared to take in order to satisfy customer needs).

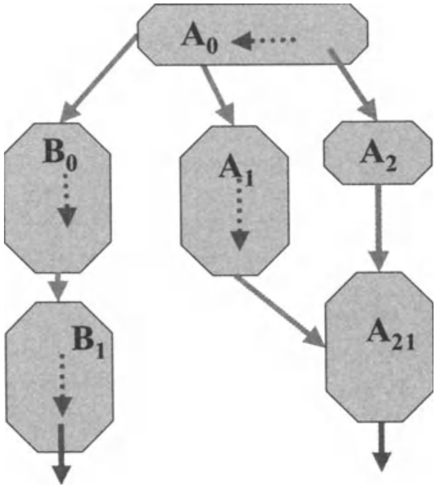


Fig. 16.2. Corresponding organisational chart (*c.f.* Fig. 16.1)

The flow of the decision framework from $B_0$to B_1 is interesting in itself (Fig. 16.3 shows its fine structure):

- The decision framework does not represent a command and control structure. It represents a standard channel of negotiation, through which $B_0$and B_1 agree on the actual objective to be achieved. As part of the job description B_1accepts B_0's authority to make a final decision. This is part of the pre-negotiation, based on which B_1'becomes attuned' to the types of requests that B_0 is expected to determine in the future. On the other hand, B_0 also accepts that B_1 cannot be voluntaristically controlled; each actual decision framework is to be passed on through a negotiation, however simple. It is also understood, that if B1 accepts the actual decision framework, then it is expected that either the task will be carried out according to the actual objectives, or B_1 will return to B_0 in a timely fashion, reporting on any obstacles. This might lead to a re-negotiation of the actual objective, decision variables or constraints.
- Once the negotiation is complete, $B_0$is also entitled to measure the performance of B_1 through performance indicators which are part of the agreement.

However simple the above structure appears, in reality few organisations follow such a protocol, and encounter difficulties and/or assess their employees

based on the wrong performance measures. After all, if an objective is not achieved, who is at fault: the person who was given the task, or the person who determined an unrealistic objective? Either or both might be case. Removing elements of the negotiation is possible under certain conditions, but it is not advisable to remove them without considering the effects.

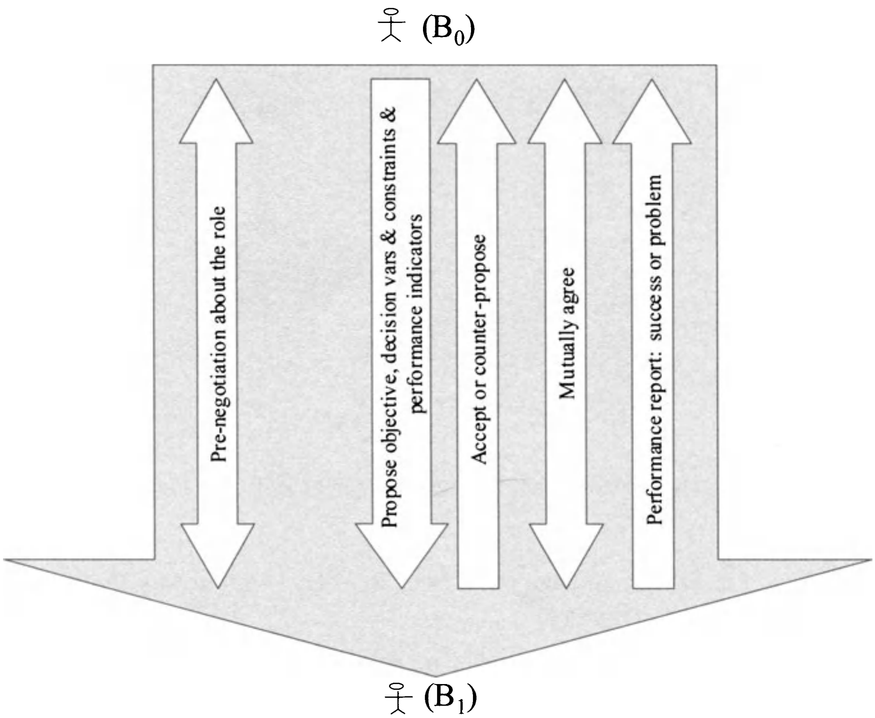


Fig. 16.3. The decision framework as a negotiation

A simple organisational chart (as in Fig. 16.2) may be useful for a small organisation, where the number of management jobs is so limited that it is not necessary to aggregate jobs into higher-level organisational units. For any larger organisation, such a simple structure would be too complex to handle. Thus the questions are: (a) in what way can one aggregate jobs into organisational units (usually on multiple levels), and (b) what are the principles that guide such an aggregation? Furthermore, whether or not such aggregation is made, (c) what are the principles that should be adhered to, so as to make sure that the mapping of management tasks to jobs is satisfactory?
For example, in Fig. 16.2 job A_{21} is a difficult one, because this person has two bosses, thus the responsibility structure is not clearly defined. In case the objectives defined by A_1 clash with the objectives defined by A_2, A_{21} must

handle conflicts that are not in its power to resolve (see also Section 16.2.3). This is only one example for possible problems with job allocation; more will be discussed below.

16.2.2 Flat vs Hierarchical Organisation

The decision hierarchy of the management system is the result of the management task at hand, and is not deterministically connected with the organisational hierarchy. Figure 16.4 shows a management system (with management tasks represented as rectangles). The same management system is implemented in Fig. 16.4a as a hierarchical organisation (with four levels of management) and in Fig. 16.4b as a flat organisation (with two levels of management).

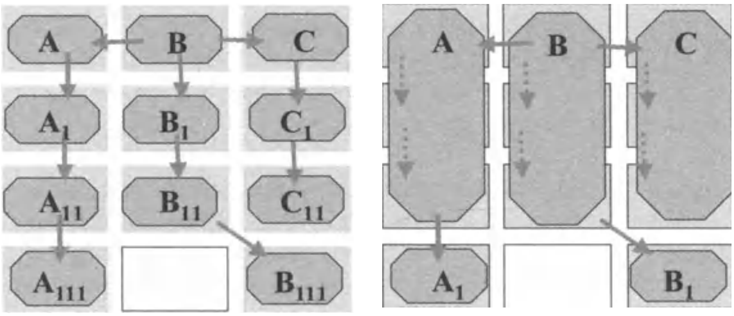


Fig. 16.4. Hierarchical organisation (a) vs. Flat organisation (b)

Aggregating management tasks into one job has limitations:

- too many tasks for one person can create information overload;
- the skills necessary to perform different management tasks are different, and one cannot create a job description that calls for a mix of skills unavailable in the company or even on the job market.

Figure 16.5 for example, illustrates an organisational structure that has jobs with a large vertical span on the resource management side. A person who takes such a job must be involved in every type of decision from strategic decisions about resources to operational management. Figure 16.5 also illustrates a problem on the strategic level: human, financial and technical considerations are allocated to different people, thus making it hard to develop a balanced resource strategy. Such separation of areas is more suited to lower level of management, but not to a strategy-making role.

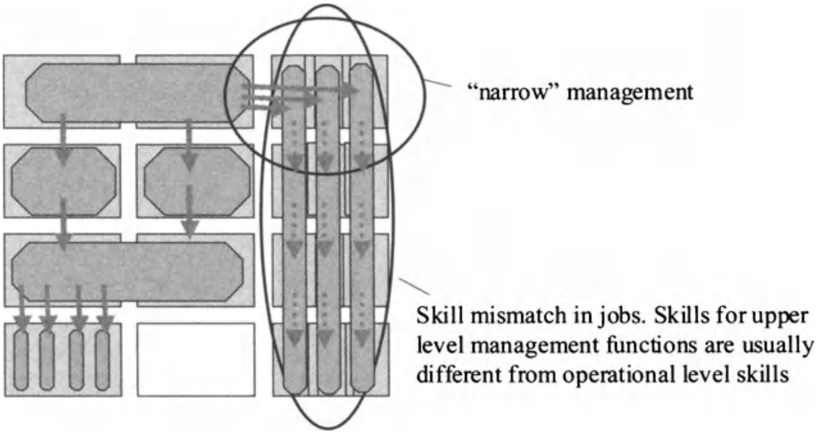


Fig. 16.5. Illustration of two problems: narrow management and skills mismatch on the resource management side

16.2.3 Conflicting Roles

Figure 16.6 shows a job allocation in which two jobs are in conflict. On the level of product strategy development B is the superior of A, while on the shorter horizon A is the superior of B. With the exception of individuals who have a long term working relationship, mutual respect and trust, and no egos, such allocation of jobs is bound to create conflict.

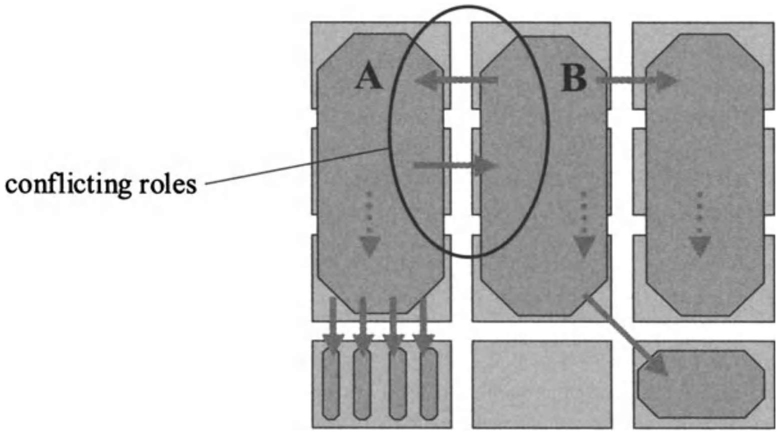


Fig. 16.6. Conflicting roles between jobs A and B

16.2.4 Over-managed Roles and Consultative Roles

Figure 16.7 shows a job allocation in which C on the strategic level does not have any decision power (it does not determine the decision framework of any other role, thus C is only employed in a consultative role). B, as C's supervisor determines the objectives for C on the tactical level[2]. In other words, C's role is either over-managed, or is deliberately kept in a consultative-only role. This type of arrangement might work if C on the strategic level is intentionally given a consultative role, in which case C is not responsible for actual decision-making. As a result C's area of strategic management has a diminished influence.

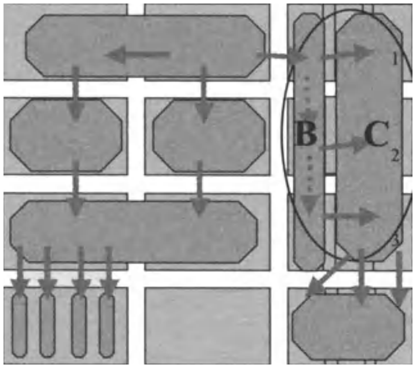


Fig. 16.7. Over-managed and/or consultative roles[4].

While such structure is often not ideal, there are situations where the given area of management is deemed to be of lesser importance on the given level. However, the organisational designer must be aware of the problem and consider whether this arrangement is valid. E.g. sometimes the organisation's realities dictate that some people of influence in the company remain in a position that is seen to be important, however the person should not be given a direct decisional power. In this way, the company can use the knowledge of that person (who might have a long-standing history within the organisation). It is still better in such cases if the consultative role is separated from the rest of the roles, e.g. roles 1 and 2 are allocated to the in-house consultant, while role 3 is given to another person. Thus C_1 and C_2 becomes a purely consultative job, while C_3 is created as a separate a job with decisional power.

[2] this type of scheme is also known as *paternalistic management*

[4] Example shown: C is not given enough authority / decisional authority

16.2.5 Aggregation of Roles into Organisational Units

As discussed in Section 16.2.1, in any larger organisation jobs need to be aggregated into higher level organisational units, such as business units, divisions, departments, teams, committees etc. This achieves two objectives:

- The organisation's design becomes less complex, because on any level of organisational design there is only one level to be considered, i.e. what is a good way to subdivide the unit into smaller units?
- Organisational units receive autonomy regarding their internal organisational structure; thus, the person responsible for the unit will be empowered, and the unit's organisation can be rearranged as necessary, resulting in a more dynamic organisation.

Much of organisational design literature treats this problem of aggregation, which is valid, but one must keep in mind that organisational design is not a substitute for the specification of the decision structure (discussed in Chapter 12). In practice, management often comes to the conclusion that the company needs to be re-organised, but omits the step of investigating the decision structure. If the decision structure is not appropriate, then no organisational design (re-shuffling of jobs) will help.

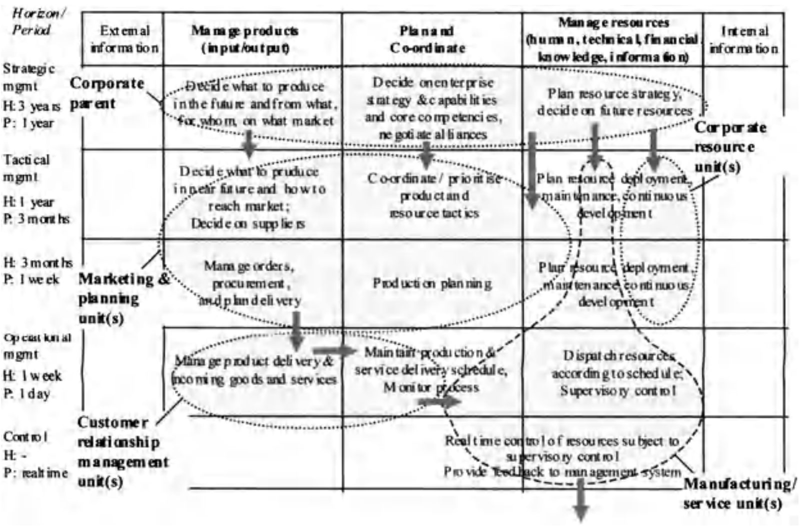


Fig. 16.8. Aggregation of management roles into organisational units (c.f. Chapter 12, Fig. 11.3

Figure 16.8 shows a possible aggregation of roles in a management system into organisational units. The corporate parent as a unit determines the com-

pany's strategy and is responsible for the level of organisational design shown in this figure, as well as its own detailed organisational design. Each other unit is treated by the corporate parent as one organisational entity, and are responsible for their own detailed organisational design.
As organisational design skills may be insufficient within some of these units, the corporate resource & corporate development unit(s) may provide a consultative role to support other units in their organisational development. In this way, the organisational units become enterprise entities on their own right.
Similarly to the identification life-cycle activity described in Chapter 2 and detailed in Chapter 7, when the mapping the overall decision structure to organisational units (enterprise entities) each entity may be recursively designed using the same methods. As part of this aggregation the relationships among these entities is defined, with the main relation being determined by the decision frameworks that connect these entities (represented as thick arrows in Fig. 16.8).
The basic method of aggregation needs to be supported by *principles* of organisational design, which are treated in Section 16.3 below.

16.3 Defining the Organisational Structure - Organisational Fit and Principles

There are many ways to create organisational units (aggregates), and a number of considerations are necessary to ensure that the organisational design fits the purpose as well as the design passes a number of tests so that that important systemic properties are not overlooked. For example, the design might be correct from the point of view of a unit's functionality, but if that design prevents the unit from developing in the future, then the organisation as a system lacks the 'evolutionary property'.
In this section the concept of *organisational fit* is defined, and the organisational design principles of Goold and Campbell (2002) are presented, extended with a brief discussion of important choices that the designer can make. These choices depend on various situational factors of the enterprise. These characteristics include internal end external factors, such as geographical distribution, being part of a network or not, relative size of partners, level of trust established with partners, homogeneity / diversity of culture within the enterprise, maturity metrics, the complexity of the operation to produce products and services, and the pace of change in the given industry.

16.3.1 Organisational Fit

Mintzberg (1979) has developed a taxonomy of organisational forms, which is based on the complexity resulting from the above factors and the pace of change in the given industry. According to this categorisation, Mintzberg defines four types of organisational forms (refer Table 16.1).

Table 16.1. Mintzberg's taxonomy of organisational forms

	Simple	**Complex**
Stable	**Machine bureaucracy** (work processes and outputs are standardised)	**Professional organisation** (skills and norms are standardised)
Dynamic	**Enterpreneurial startup** (direct supervision)	**Adhocracy** (mutual adjustments)

The basic mechanisms of co-ordination that suit these types of organisation are different.

In a *machine bureaucracy,* it is possible to define standard work processes (often procedures). In all other organisational forms, this is either impossible or limited. The contents of decision frameworks could be objectives that instruct the supervised role to perform some predefined procedure, and only the parameters of these procedures would change.

In a *professional organisation* objectives could only be defined through referring to the *type* of output that is desired, leaving the judgement of the best possible output to the subordinate role. The defined objective would not refer to predefined procedures (or only to some, but not all), and the processes followed by subordinates would be guided by agreed principles. The control of such an organisation relies on defined professional skills of people who fill the management roles.

In an *enterpreneurial startup* organisation roles can only be defined to form a flat organisation, where management takes a direct supervisory role in the operations, thus the decisional structure can only be defined in broad terms (refer for example to Fig. 16.6b). The decisional structure is basically hidden – which does not mean that people in such roles do not have to attend to strategic, tactical and operational decisions; it only means that the decisions are taken for themselves rather then for subordinates. Also, the decision frameworks that flow between managers of an enterpreneurial organisation are better populated with objectives that do not refer to standardised procedures, but instead refer to agreed-on principles and policies (similarly to professional adhocracies); at the same time, the decision frameworks in such an organisation rely on negotiations more then in other forms (refer to Fig. 16.1).

In an *adhocracy*, similarly to professional organisations, decision frameworks would refer to *types* of outputs where the actual output is determined by the subordinate role. The setting of objectives at the same time would be defined in a negotiation, similar to enterpreneurial startups.
On a high level, the subdivision of the organisation may use five types of components (Mintzberg , 1992):

- the strategic apex (responsible for strategy making and typically consisting of the CEO and a board of directors);
- a technostructure (incorporating planning, training and various analytic tasks);
- support units (performing functions that are not directly related to the mission of the company, but are needed for the organisation to function);
- the middle line (typically line managers for marketing, sales and operations) and
- the operations itself (the units that perform the company's mission, i.e. carry out the production and service tasks).

Today (as discussed in Section 16.2.1 and in Chapter 12), the dynamism of the environment is becoming an increasingly determining factor for companies, and with this the role of negotiation in objective setting is becoming more important then in the past. Thus, modern organisational forms are becoming more network-like, where negotiated collaboration and co-ordination plays a bigger role then in traditional organisational forms (Pettigrew and Fenton , 2000).
The concept of organisational fit (Donaldson , 1995) does not argue for a unique (or 'best') design; it only stipulates that the organisational design must pass the tests that determine whether it is a good design or it is not suitable. The practical manager would certainly find it difficult to keep up-to-date with new organisational theories, and even if this is possible, in addition to analytic theories it is necessary to develop a practical design methodology that not only explains the properties of existing organisational forms, but also gives some form of guidance for organisational synthesis. Goold and Campbell (2002) describe an organisational design framework, with criteria for testing whether the organisational design is fit for the purpose. The framework can also be used as the basis for a practical synthesis process[5].
Goold and Campbell's Framework (refer to Fig. 16.9) consists of:

- a set of "fit drivers" – i.e. those tests that are mandatory for an organisational design to be suitable for the purpose; and
- a set of organisational design principles – i.e. those principles that should be followed for the organisational design to be of good quality.

[5] This does not mean that the synthesis process is a step-by-step procedure; such a result cannot be expected, because for any given purpose there are many suitable organisational forms

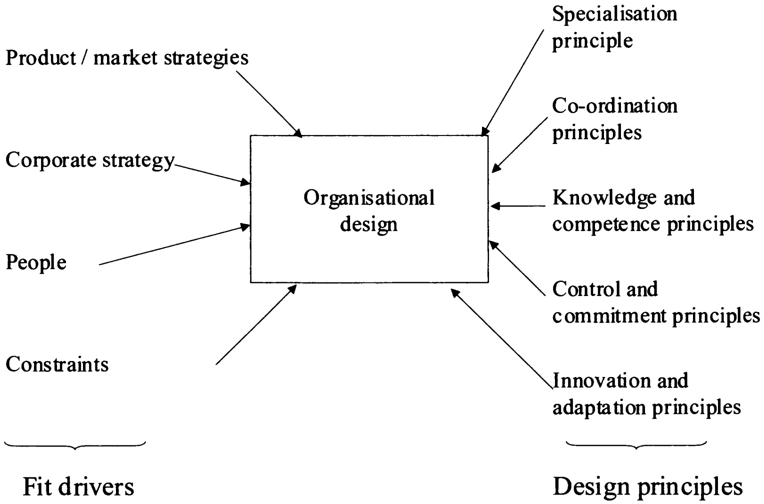


Fig. 16.9. Goold and Campbell's Framework for Organisational Design

Goold and Campbell explain the four drivers of fit as follows:

- "Product market strategies: how the company plans to win in each product-market area it choses to compete in;
- Corporate strategy: how the company plans to gain advantage from competing in multiple product-market areas;
- People: the skills and attitudes of individuals who are likely to be available to work within the organisation; and
- Constraints: the legal, institutional, environmental, cultural, and internal factors that limit the choices of design" (Goold and Campbell , 2002)

Thus, in any organisational design effort the proposed design must be measured against these fit drivers, and if the design does not pass the test, it must be modified.

16.3.2 Design Principles and Choices

One important skill that differentiates experienced designers from novices is their thorough understanding of design principles. Given the endless variety of possible designs the designer is faced with too many choices, and to make progress toward the design goal some form of direction is needed. Various design schools list different principles of good design, but they mainly differ in the way these are categorised. A long list of elementary principles has the advantage that the principles are simple and concentrate on single concrete issues. This has the advantage that the principles are easy to understand.

However, a list of forty principles is too long to keep in mind, and gives less orientation to the designer then a short list – to be precise as long as our human capacity allows us to keep in mind at any one time (Miller , 1956). In this regard, Goold and Campbell's list of five principles (discussed below) is suitably short, while still rich enough to not being overwhelmingly abstract, and can be used to either test a design or to direct a design process. These five principles are discussed below, and are also listed on the right hand side of Fig. 16.9.

The *specialisation principle* draws the designer's attention to the fact that decisions are to be made about where to place the boundaries of organisational units. For example, if the company's product-market strategy is to establish a widely diversified enterprise, then it is likely that the skills involved in different products are different. Thus, any organisational unit that groups individuals with similar skills is likely to be smaller than in an enterprise that has a single product strategy. As a result, "unit boundaries should be defined to achieve the most important benefits available from specialisation" (Goold and Campbell , 2002).

The *co-ordination principle* states that it is easier to co-ordinate activities if they fall within the boundaries of the same organisational unit. Thus, if the decisional structure is such that all decision centres within the organisational unit's boundaries receive their decision framework from within the unit, then it is easier to maintain co-ordination. Unfortunately, this is not always achievable, because either (a) important decisions may have to be made outside of the organisational unit, or (b) strong information links must be present between two or more units, and it is well known that (notwithstanding the beneficial use of modern information and communication technologies) information flow across unit boundaries is always more difficult than information flow within unit boundaries.

Summarising the two principles above, it becomes apparent that the greater the number of organisational units, the harder it becomes to co-ordinate the entire company's activities. Therefore, these two principles together imply that a trade-off is to be made between the benefits of unit size and difficulty of co-ordination. Since units need a certain level of autonomy (especially if a difficult technology, or a very special geographic or cultural area is involved), the organisational designer may take this into account and favour the need of autonomy over the benefits of specialisation. For example, while a university may decide to centralise all information technology management under one corporate service (thus creating the benefits of specialisation), it might not extend this unit's responsibilities to the computer science department, which is not only the user but also the developer of new technologies, and as such has special needs – e.g. frequent reconfiguration, experimentation, involvement of staff in the development of new services, etc.

The *knowledge and competence principle* states that "responsibilities should be allocated to the person or team best placed to assemble the relevant knowledge and competence at a reasonable cost" (Goold and Campbell , 2002).

Thus, if the organisational unit does not have the capacity to gather all necessary knowledge to perform its operations effectively and efficiently, the unit will not be self-standing and decision-making will need the involvement of outside units (creating co-ordination problems and hurting the autonomy of the unit). Placing the missing management role within the unit can solve this problem. The result of observing this principle tends to create a flatter organisation, in which only those decision functions are retained by higher level management that add value to the organisation.
For example, consider the decision functions in Fig. 16.6a. If the link between tactical and operational management (e.g. between A_1 and A_{11}) is a decision framework that requires frequent and involved negotiations between the two levels, this is a sign that there is a lack of competencies in operational management – due to tactical level decision functions being different in nature from those necessary on the operational level. In this case, higher level management (such as a corporate parent) would best perform functions A, B and C, and units would be formed as shown in Table 16.1, with tactical management being part of the lower level units. This is consistent with the resource-based view of management (refer to Chapter 4), which identifies knowledge and competencies as two of the most important resources of the company.
The problem is most difficult if an important part of the knowledge required is tacit (not externalisable in form of explicit models). Goold and Campbell (2002) propose a strong test for keeping decisional power in higher-level units (such as a corporate parent): "are all levels in the hierarchy and all responsibilities retained by higher levels based on a knowledge and competence advantage?" (Goold and Campbell , 2002).

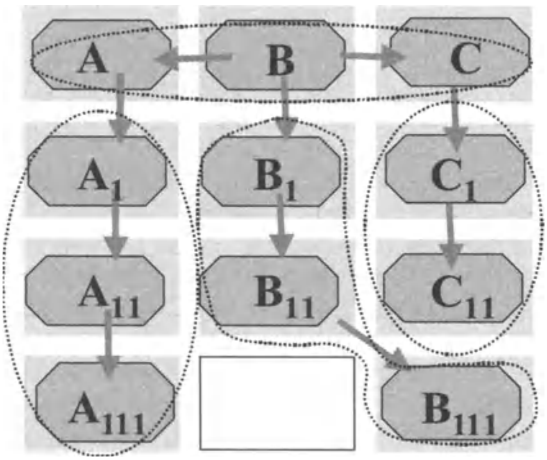


Fig. 16.10. Organisational design observing the knowledge and competence principle

The *control and commitment principle* states that "units should be formed to facilitate effective, low-cost control and high commitment to appropriate goals" Goold and Campbell (2002). The boundaries between units in the decisional hierarchy are in practice between the head of the lower level unit and a superior manager. These boundaries are important because this is the link through which the higher-level unit controls the lower level one, and through feedback, the higher-level unit can measure the performance of the lower level. In setting organisational boundaries it must be observed that the more control the higher unit exercises (through tightly chosen decision frameworks), the more difficult is becomes to motivate the lower level, because its lost autonomy de-motivates the lower unit's management. On the other hand, less tight control means that higher-level management needs appropriate feedback (in form of performance indicators) to measure the success or failure of this relationship.
Performance indicators do not necessarily measure the success of the lower level unit – it is possible that the higher level, though inappropriate interference with the measured unit's decisions, artificially blocks optimal performance. If an organisation design proposal does not allow for the managed unit to be motivated, or if no clear performance indicators can be designed, then unit boundaries might need to be changed.
The *innovation and adaptation principle* is the last of the five, stating that "organisations should be structured so that they can innovate and adapt as uncertainties become clarified and environments change" (Goold and Campbell , 2002). To test whether this principle is adhered to, it is to be investigated whether the design allows flexibility, in terms of developing new strategies that lead to necessary changes in the future.
The source of strategic innovation does not necessarily come from those who take the strategic manager's role. Organisational structures that have the information channels and responsibility structures which are receptive of new ideas, and give the authority to lower levels to experiment with new solutions are more likely to generate emerging strategies, those that may later become adopted as part of a corporate strategy. Such structures are most important in organisations that fall into the professional organisation, enterpreneurial startup or adhocracy categories (refer to Fig. 16.2).

Table 16.2. Co-ordination mechanisms suited to Mintzberg's four organisational forms

Organisational Form	**Coordination Mechanism**
Machine Bureaucracy	Standardised procedures and outputs
Professional Organization	Standardised professional skills and norms
Entrepreneurial Startup	Direct supervision and control
Adhocracy	Mutual adjustment of ad-hoc teams

The organisational designer has a number of other aides that suggest (and not only test) suitable designs. For example, Keidel (1995) has identified a simple but powerful model that makes design choices explicit. In any organisational design issue, Keidel proposes to consider the placement of the solution into a three-axis triangle: with autonomy, control, and co-operation being in the three corners. Given an organisational design issue, e.g. considering the organisational structure (layout), one can decide whether the structure will be based on autonomy, on control, or on co-operation or a compromise between these extremes. Choosing any of these as the guiding theme will result different organisational structures, but it is also possible to mix these. Keidel proposes that trying to create a design that combines all three is not a weak (indecisive) design choice (unless equal weight is given to each of the three design criteria, as shown in the center of Fig. 16.11). Many design issues can be characterised in this way, and Keidel's book contains in its appendix a long list of reference models through which the organisational designer can select an appropriate style. After intuitively making a design choice on the basis of Keidel's patterns, the result must still be tested through the organisational design fit test as well as the good design principles tests as described in this Section.

16.3.3 Organisational Design Method

The actual process of organisational design may start from any of the following three points (Goold and Campbell , 2002):

- The current organisational design – with the aim of finding deficiencies and ultimately coming up with a better design;
- A proposal for a new design – where the new design (most likely a modification of the existing one) is intuitively put forward by management, and this design is to be analysed, potentially changed and improved;
- The definition of design criteria that a new organisation will have to achieve – where the objectives of the organisational change are known, but no new design is yet proposed (e.g. to create a brand new design for a new venture).

If the third of the above cases applies, then management must subsequently propose various design alternatives for testing. Design alternatives can be generated by first recognising the place of the organisation in Mintzberg's organisational taxonomy (refer to Table 16.1), subsequently using the organisational design patterns of (Keidel , 1995), and of course, employing plenty of intuition and management experience.

While many organisational design efforts start from a proposal based on another company's structure, this initial proposition must be treated as a design proposal – what works for one company may not work for another, and the tests below will reveal such weaknesses.

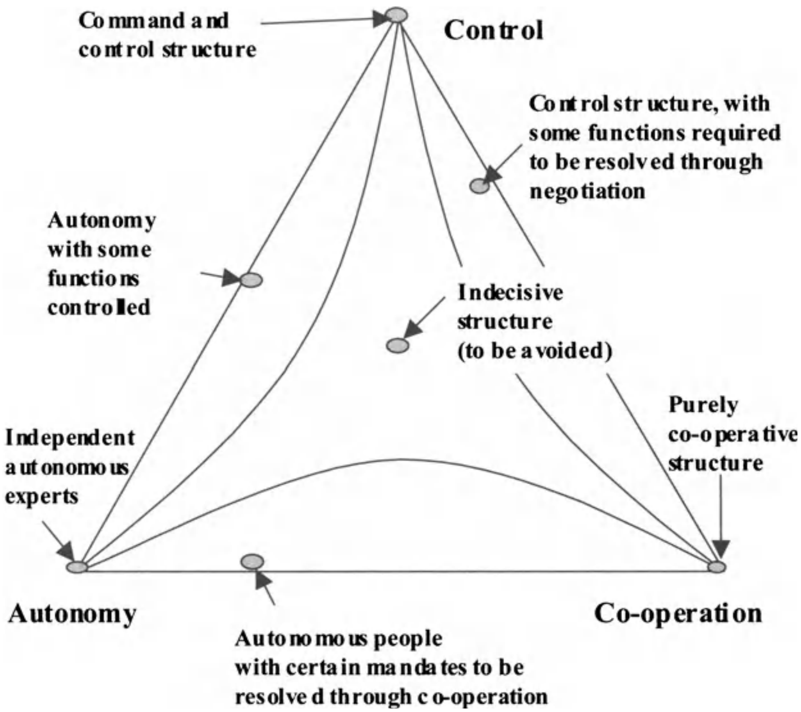


Fig. 16.11. Keidel's triads: control, cooperation and co-operation (on the example of organisational structure)

Any of the above propositions ends up with a statement about the new organisational design. The design must detail the structure of the organisation, the responsibilities and authorities of the units, and describe the nature of decision links that relate these units to one another.
This statement must be clarified, through consulting with the proposers, and the proposals must be thus tested:

- test for organisational fit – to ensure that the proposed design (or design alternatives) passes the four tests of fit, as described in Section 16.3.1.
- test for good design principles as described in Section 16.3.2.

The organisational design fit and the good design principles tests are somewhat different. It is not acceptable for the design to fail any of the fit tests. However, the results of the good design tests are likely to be less clear, because these criteria often call for compromises: if the design is excellent from the point of view of one test, is may not be as good from the point of view of another test. For example, to satisfy both the specialisation principle and the

co-ordination principle the designer may have to select a trade-off. Thus the results of design principle tests are likely to suggest smaller design modifications.

If the starting point was a statement of design criteria, then the designer may rank the proposed alternatives, and eliminate those that rank the lowest ones until only one proposal remains.

The fit test and good design test are the core of this process; the other elements of this process are neither very new nor surprising. However, the tests are an often overlooked element, and without them management may end up with surprises and costly corrections.

Finally, when management is satisfied with the new design the chosen design must be reviewed and communicated to all involved parties.

16.4 Conclusion

This Chapter has reviewed organisational design as the process of aggregating management functions into organisational units, and has discussed a design process involving tests for organisational fit and good design principles. The application of these principles ensures a better chance of producing successful organisational design results.

References

Donaldson, L (1995) American Anti-Management Theories of Organization: A Critique of Paradigm Proliferation. (Cambridge Studies in Management, #25) Cambridge, UK : Cambridge University Press

Goold, M., Campbell,A. (2002) Designing effective organizations. San Francisco : Jossey-Bass.

Keidel, R.W. (1995) Seeing Organizational Patterns. San Francisco : Berrett-Koehler Publishers

Miller, G.A. (1956) The magical number seven, plus or minus two: some limits on our capacity for processing information. The Psychological Review. Vol. 63. pp81-97

Mintzberg, H.(1979). The structuring of organisations. Englewood Cliffs, N.J. : Prentice-Hall

Mintzberg, H. (1992) Structure in Fives: Designing Effective Organizations. Englewood Cliffs, N.J. : Prentice Hall

Pettigrew, A.M., Fenton, E.M. (Eds) (2000) The innovating organization. London: Sage

Thompson, S. (1967) Organizations in action. New York : McGraw-Hill

Tushman, M., and Nadler, D. (1978) Information Processing as an Integrating Concept in Organizational Design. Academy of Management Review 3(3) pp613-624.

17

APPLICATION REFERENCE MODELS AND BUILDING BLOCKS FOR MANAGEMENT AND CONTROL

Michael Rosemann

Queensland University of Technology (Australia)

17.1 Introduction

Reference models are generic conceptual models that formalise recommended practices for a certain domain. Often labelled with the term 'best practice', reference models claim to capture reusable efficient state-of-the-art practices. Thus, a more realistic label would be 'better practice' or often even 'common practice'.

The depicted domains can be very different. They can range from selected functional areas such as Accounting or Customer Relationship Management to the scope of an entire industry sector, *e.g.* mining. A domain can also cover specific projects such as certification, or the implementation of an Enterprise System. Reference models are often derived from previous individual conceptual models through a continuous evaluation and consolidation of these models. Application reference models help, for example, a consulting company to continuously consolidate their knowledge gained within similar projects.

An organization may also declare the models describing one subsidiary as the internal benchmark. Thus, an existing conceptual model can get the status of a reference model. This practice can be observed especially in global organizations that roll-out the business blueprint of one location to all their subsidiaries worldwide. These models are also called prototypical models (Bernus and Nemes, 1996). Sophisticated modelling tools allow the use of a common global database and the management of model variants.

The main objective of reference models is to streamline the design of enterprise-individual (particular) models by providing a generic solution. Reference models aim to reduce the costs of designing models, which facilitate the management and control of an organization. They accelerate the modelling process by providing a list of potentially relevant business processes and structures. Such models have only to be individualised, rather than designed from scratch. Consequently, reference models can also be regarded as a source of ideas and can serve as a base for discussions. Partial models available for a specific modelling task may form a library or repository of potentially useful models intended

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

to increase the modelling efficiency. These (ideally 'plug and play') models are also called Partial Enterprise Models (PEMs) in the terminology of the Generalised Enterprise Reference Architecture and Methodology (GERAM) (ISO/TC184/SC5/WG1, 1999)

Reference models can be very comprehensive in terms of scope and quantity of the models. Every application of a reference model requires a process of selecting the relevant parts of the model (*e.g.*, tailoring) and in the most cases further developing and extending (*e.g.* specialising) these model extracts. From this perspective, reference models can be regarded as structured building blocks. However, current reference models are designed with conventional modelling techniques that are not designed for the purposes of selection and individualisation. In general, they are based on the ready-to-run-assumption that the building blocks can be utilised one-to-one for the purposes of the individual organization. This missing capability to individualise reference models can be regarded as a major shortcoming.

This Chapter is structured as follows. The next section will differentiate three types of reference models. The main part of this Chapter will propose new concepts that allow the selection and individualisation of building blocks based on the explicit modelling and configuration of alternatives in reference models. The Chapter will end with a brief conclusion and outlook.

17.2 Types of Reference Models

17.2.1 Criteria for Differentiation

Reference models were defined as generic conceptual models that formalise recommended practices for a certain domain. This Chapter only focuses on models that are of relevance for the requirements engineering phase. Reference models for the system design or the implementation phase or models for the standardisation of communication protocols will not be discussed here.

Another class of reference models that is not within the scope of this Chapter describes reference methodologies. These reference models specify on a higher level of abstraction constructs such as the core object types and relationship types of a data modelling technique. These models can be characterised as reference meta-models.

A core criterion to differentiate reference models is based on the *view* of the reference model. This separation of a reference model helps to reduce and manage the complexity of these models. Popular views supported by reference models are data, business processes, organizational structures, resources, enterprise architectures, functions, objects, etc. or any combination of these views. If a reference model covers more than one of these views it has to be distinguished between models that take inter-relations between these models into account and those that do not support any inter-model relationships.

In the latter case, the reference process model, for example, can be individualised independently from the reference data model. This is currently the dominating case, which easily leads to inconsistency between the views.

Closely related to the view of a reference model is the *purpose* that is supported by a reference model. With the increased popularity of business modelling, a wide and quite heterogenous range of purposes can motivate the use of a reference model. The list of purposes includes software development, selection of off-the-shelf-solution, configuration of Enterprise Systems, workflow management, documentation and improvement of business processes and of the overall management and control structures, education and user training, auditing and certification, benchmarking, and knowledge management to name the most popular purposes. The development of a reference model has to clearly consider the supported purpose(s) as the related requirements vary significantly. This has to be reflected among others in the views, the modelling techniques, the object types, the relationship types and in the depicted domain.

In this context it is important that reference models can also be differentiated regarding the *user group* they address. The potential user of a reference model can be positioned between the extreme of being a sophisticated business analyst with comprehensive modelling experience and a novice, who approaches reference models for the first time. Again, the addressed user group has to be reflected within the modelling techniques, symbols and supporting tools.

The combination of purpose and user is called the *perspective* on a model (Rosemann, 2000b). A current challenge for reference models is exactly the support of multiple perspectives, which is a pre-requisite for the comprehensive and cost-effective use of a reference model in an organization.

In the following, a further and more comprehensive differentiation is made based upon the *domain* that underlies the reference model.

17.2.2 Industry Reference Models

Industry reference models are designed for the purposes of capturing recommended practice in a certain industry sector or in parts of it. These models are in general designed by academics and/or consultants. The focus of these models is nowadays to describe the business processes. They facilitate especially the exchange of industry-specific knowledge. This includes not only recommended practices, but also helps to identify potential business processes first of all. Industry reference models can serve educational purposes as well as they facilitate the explanation of the characteristics of an industry.

A comprehensive example for an industry reference model is the generic model for manufacturers developed by Scheer (1998). This model is populated for the data, function and process view. In a similar way, Becker and Schütte (1996) developed a model for retailers. Silverston (2001b) presents different reference data models for specific industries such as manufacturing, telecommunication,

health care, insurance, financial services, professional services, travel and e-commerce. A reference model for IT service management covering areas such as service support and service delivery is documented in the IT Infrastructure Library (ITIL) (CCTA, 2000, 2001).
Many industry reference models can be found in practice. The IT consulting company Logica developed in the Netherlands a comprehensive reference process model for the telecommunication industry, which can now be used by all their subsidiaries worldwide. KPMG used the ARIS Toolset (Davis, 2001) in order to design a reference model for the insurance and the utility industry. The IDS Scheer AG provides among others reference models for the industry sectors mechanical engineering, plant engineering, consumer goods industry or paper industry.
The term 'industry reference model' does not require that the model has to be only of relevance for one particular industry. The Supply Chain Operations Reference model SCOR (http://www.supply-chain.org) fits also very well in this category. Silverston (2001a) presents reference data models for selected areas, such as people and organization, ordering, shipment, invoicing or accounting.
Industry reference models can be differentiated along the following main criteria

- Scope of the model (e.g., functional areas covered);
- granularity of the model (e.g. number of levels of decomposition detail);
- views (e.g., process, data, objects, organization) that are depicted in the model;
- degree of integration between the views;
- purposes supported;
- user groups addressed;
- internal or external (commercial) use;
- availability of the model (e.g., paper, tool-based, Web-based);
- availability of further textual explanation of the model;
- explicit inclusion of alternative business scenarios;
- existence of guidelines on how to use these models;
- availability of relevant quantitative benchmarking data.

Most industry reference models are utilising modelling techniques that are not designed for the purpose of reference modelling. In most cases they are using an ER-approach for the data models and popular process modelling techniques such as the Event-driven Process Chains (EPCs)for the process models.
The data models of typical industry reference models include approx. 300-400 entity types and the process view often covers 80 and more business process models. Because of their enormous complexity, industry reference models do in most cases not depict alternatives, but are based on the ready-to-run-assumption. Though they often come with textual explanations, guidelines

for the model use and configuration and benchmarking data are usually not provided.

Industry reference models can be regarded as mission support models as they specify the individual processes and structures of a specific industry. The e-Telecom Operations Map (eTOM), a framework for the information and communication services industry will be discussed later on as a more comprehensive example for such an industry-specific reference model. eTOM is currently developed by an international consortium of communications services providers and their suppliers. eTOM combines the TOM framework with business partner relationship models. It aims to provide operational guidance for the design of business processes and inter-process information flows specific for this industry. Furthermore, it addresses the design of an appropriate IT infrastructure and the development of a market and products for integrating and automating telecom operations processes. The following objectives of the eTOM Business Process Framework clearly characterize this model as an industry reference model (Telemarketing Forum, 2001, p. 32):

- "An 'industry owned' common business process framework.
- Common definitions to describe processes of a service provider.
- Agreement on the basic information required to perform each process, sub-process and process-activity, *i.e.* sufficient high level information to serve as the starting point for business requirements and information model development, and the satisfaction of those requirements through industry agreement in business-aware contracts, shared data model elements, and supporting system infrastructure and products.
- A process framework for identifying which processes and interfaces are in most need of integration and automation, and most dependent on industry agreement."

An explicit goal of eTOM is to "create a library of process flow examples" (Telemarketing Forum, 2001, p. 39). The following details represent the status of eTOM Version 2.5 and are not a final version. The most current information can be found at http://www.tmforum.org.

Fig. 17.1 provides an overview of the Level 0 of the eTOM Framework. The two main vertical end-to-end groupings into Strategy and Operations become obvious. The Operations process area is regarded as the core of eTOM. Five key functional areas are depicted as horizontal layers. Furthermore, the figure shows the internal and external entities.

On level 1 (called CEO level view), these areas are depicted in further detail. For example, Operations is differentiated according to the lifecycle of the telecommunications product (service) into Operations Support and Readiness, Fulfilment, Assurance and Billing, of which the last three processes form the Customer Operation (also Priority) Processes. The Strategy grouping covers processes that drive and support Operations Support and Readiness and the Customer Operation Processes.

Using hierarchical decomposition, level 2 includes a further detailed view of the reference process models. The end-to-end process Assurance is, for example, further detailed into Customer Relationship Management (Customer Interface Management, Problem Handling, Customer QoS/SLA[1] Management, Retention & Loyalty), Service Management & Operations (Service Problem Management, Service Quality Analysis, Action & Reporting), Resource (Network, Computing and Application) Management & Operations (Restoration, Data Collection, Analysis & Control), Supplier/Partner Relationship Management (Interface Management, Problem Reporting & Management and Performance Management). The detailed process flows are described on up to four levels.

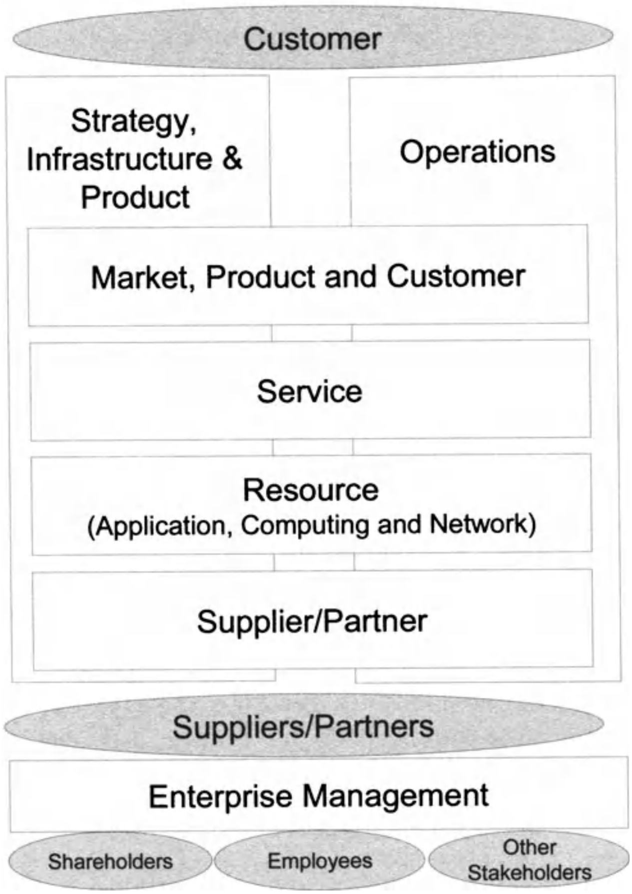


Fig. 17.1. eTOM Business Process Framework – Level 0 Processes

[1] Quality of Service / Service Level Agreement

17.2.3 Project Reference Models

Project reference models are generic guidelines for the execution of specific projects. These reference models typically come in the form of reference process- and organizational models. They capture the sequence of the required tasks and recommendations for the project organization. The core is the reference project plan, which can be interpreted as one comprehensive business process model. More sophisticated project reference models differentiate between the priority or necessity of functions or assign generic organizational units to the functions. These models are designed for projects such as the implementation of Enterprise Systems, Business Reengineering projects or certification. Figure 17.2 shows as an example an extract of the IDS project reference model for the implementation of a Balanced Scorecard.

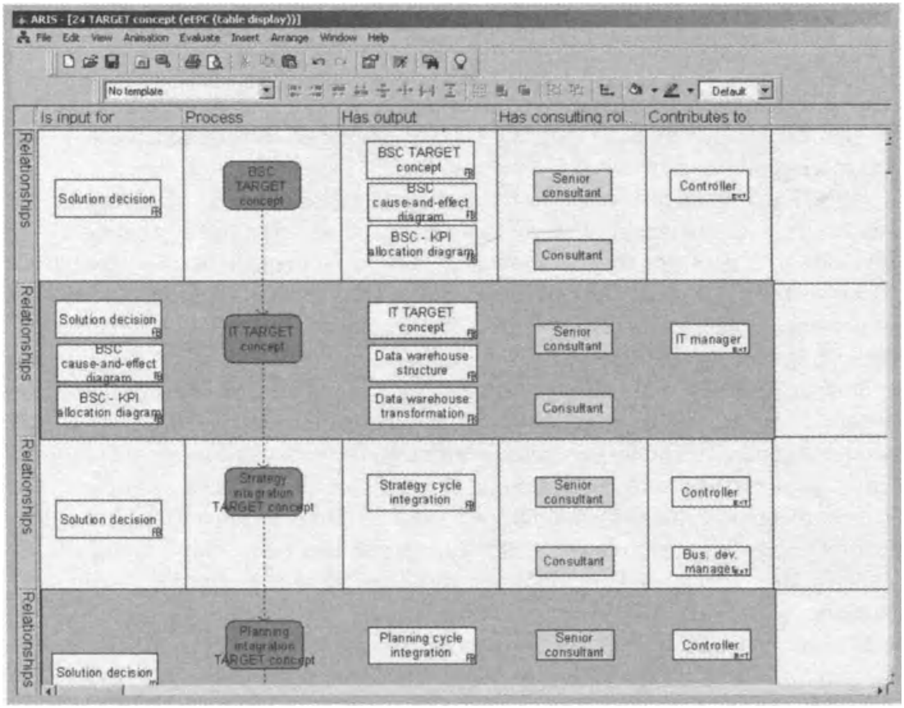


Fig. 17.2. Example of a project reference model

17.2.4 Application Reference Models

In addition to general domains of enterprises, the term *reference model* is also used for models describing the structure and functionality of 'software solu-

tions' (Curran and Keller, 1998) – applications supporting the various management processes in an enterprise. Software systems that implement such reference models are generally called Enterprise Resource Planning (ERP) systems (therefore in terms of GERAM ERP systems are a collection of Enterprise Modules, from which particular operational systems can be configured). Reference models are well-structured semi-formal descriptions of particular applications. These models support the evaluation of the appropriateness of the software. Furthermore, they facilitate the implementation of the software and the related user training (Rosemann 2000). Depending on the importance of the software, they also can be used as initial models for the design of particular enterprise-specific models (Gulla and Bratsevik, 2000).

Application reference models are significantly different to industry- and project reference models as they correspond to an existing off-the-shelf-solution that supports the functionality and structure described in the reference model. Thus, application reference models reflect more software-related constraints than industry reference models. Consequently, they are typically on a lower level of abstraction and use more software-specific terms compared to an accepted industry-wide terminology. As they are focused on the areas supported by the software, they are claiming a lower degree of comprehensiveness than the more generic industry-reference models.

Application reference models were often not the conceptual platform for the design of these systems, but were actually designed *after* the system had been developed. Therefore, they document, post-implementation, the capabilities of the system. These models are available in a wide range of model types such as process models, function models, data models or object-oriented models. Regarding the organizational model, application reference models provide system organizational units that depict the core infrastructure of an organization (legal entity, factory, warehouse, distribution channels, etc.). Applications that are documented in the form of such reference models can support certain functional applications such as Financial Accounting or Materials Management, inter-enterprise applications such as Customer Relationship Management or Supply Chain Management or entire Enterprise Systems. These functions are often further specialised for different industry sectors including automotive, retailing, high tech, etc.

It is usually possible for a user to refer to the relevant part of the online documentation and (at the lowest level of abstraction) even to the individual transactions. It has to be stressed that these models are designed for both the end users of the application and for the implementation team. End users will benefit from these models as they can comprehensively and quickly inform themselves about the related software functionality. These reference models are in most cases part of the software and do not have to be purchased separately.

As these reference models reflect the comprehensiveness of these applications, they tend to be very complex and much larger than industry reference models. For example, one of the most comprehensive models is the SAP reference

model: its data model includes more than 4000 entity types and the reference process models cover more than 1000 business processes and c-Business[2] scenarios. The two main views of the SAP reference model will be discussed in the next section. Most other market leading ERP vendors have also produced such reference models. An overview of the Baan reference model, for example, is provided in (Verbeek, 1998).

17.3 An Example Application Reference Model: SAP

17.3.1 The SAP Reference Process Model

Foundational conceptual work for the SAP reference model had been conducted by SAP AG and the IDS Scheer AG (formerly known as IDS Prof. Scheer GmbH) in a collaborative research project in the years 1990-1992. One aim of this project was to develop a modelling technique that depicts SAP-supported processes in a reasonably intuitive language. The outcome of this project was the process modelling language called Event-driven Process Chains (EPC, in (Scheer, 1998)), which has been used for the design of the reference process models. EPC also became the core modelling technique of the modelling framework called Architecture of Integrated Information Systems (ARIS - in (Scheer, 2000)). An example for a reference process model in the EPC notation is given in Fig. 17.3.

The structure of the SAP reference model comprises five levels of abstraction. The first two levels are the Business Process area (*e.g.*, Financial Accounting, Payroll, Procurement) and the Company Process area (in Procurement, for example, Procurement of Materials and Services, Internal Procurement, Procurement via Subcontracting). These levels are depicted through functions that describe the core structure of the business processes without a detailed specification of the actual control flow. Alternatively, process selection diagrams provide in the form of a matrix more detailed insight into the core alternatives within a business process scenario. Each column represents one scenario (*e.g.*, Procurement of Materials and Services and Internal Procurement). The functions within these scenarios are listed in the rows. Thus, it becomes obvious what functions are embedded in what business process. On the third level, the individual scenarios are described as Scenario EPCs, describing the control flow. Each function in these EPCs is further detailed on the fourth level, as another EPC. In the example of Procurement of Materials and External Services this decomposition would comprise functions such as Purchase Requisition, Purchasing (see Fig. 17.3), Goods Receipt, Service Entry Sheet, Warehouse/Stores and Invoice Verification. This level is called Process Group EPC. On the lowest process level, three different model types are positioned. A Process EPC describes either the detailed process or just

[2] Collaborative Business

the list of functions. These functions now correspond to transactions in the SAP application. A Function Allocation Diagram System assigns further object types to the individual functions of a Process EPC. This includes input data, screens, system organisational units (*e.g.*, Business Area, Plant, Distribution Channel), and roles (*e.g.*, Invoice Verification Clerks, Logistics Manager, Transport manager). A Role Allocation Diagram depicts the assigned screens (including the transaction codes) for each role.

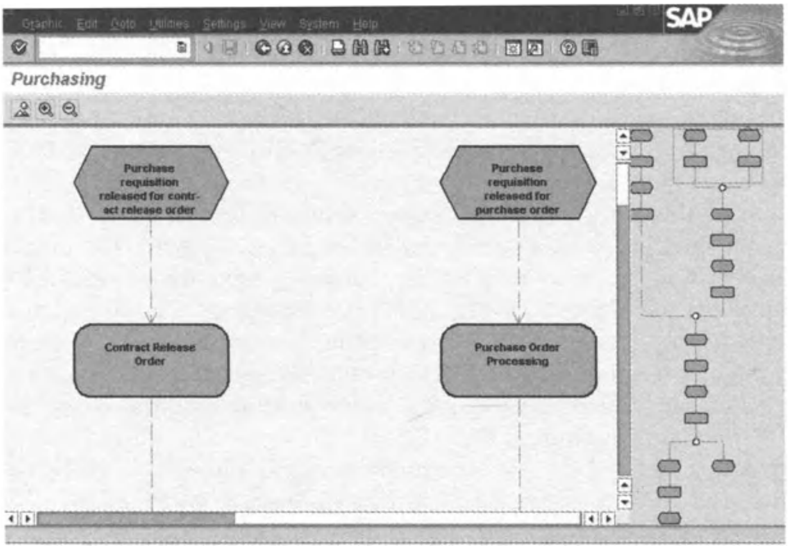


Fig. 17.3. Extract from a process reference model in SAP R/3

Recently, a technique called Collaborative Business Maps has been developed for the representation of more than 150 generic-, as well as industry-specific e-Business processes. These models describe in swim-lane notation the interaction between business partners, including the involvement of marketplaces. An exceptional feature of these models is the provision of benchmarking data including the data sources. Though this type of information is desired in most reference models, it can hardly be found in any other reference model. The e-Business Maps are available at http://www.sap.com/c-business.
The SAP reference model supports a variety of views and associated models, out of which the reference process model can be regarded as the core . The reference process model also has an established position in the official SAP-implementation methodology called Accelerated SAP (ASAP).

The SAP reference process model is available in three ways.

- The reference model can be principally accessed by all SAP users as part of the SAP R/3 system. Process models can be studied on different levels of abstraction for areas such as procurement, production, sales and distribution, or financial accounting. These models can not be further individualised (specialised) or changed. Thus, this part of SAP R/3 has the character of a semi-formal documentation of the SAP-software. Figure 17.3 shows an excerpt from the reference model for purchasing.
- As part of the ValueSAP (http://www.sap.com/valuesap) implementation tool, project members can navigate through SAP reference process models independently from an SAP R/3 installation. This is indispensable in the pre-implementation phases of an Enterprise Systems project. Again, the models in the so-called Diagram Explorer can not be modified (view only). However, it is possible to maintain further information such as the number of processed documents per period or the average processing time for every individual business process
- The reference process model can only be modified if third-party modelling tools such as the ARIS product suite are used (Davis, 2001). In ARIS, models can be deleted, changed or extended with organizational units, documents, etc. This individualised information is input to ValueSAP and determines the scope of the implementation project. SAP transactions and the corresponding online help can be called from ARIS. Vice versa, it is possible to import the relevant process models into SAP R/3.

17.3.2 The SAP Reference Data Model

Unlike the reference process model, which addresses project members and end-users, the reference data model[3] in SAP R/3 facilitates the technical implementation, and is available as a part of SAP R/3.

In general, ERP reference data models summarize on the conceptual level the data structures of the system. They are typically based on modelling techniques that extend the classical Entity-Relationship approach suggested by Chen (1976) as they include, for example, the generalization-specialization relationship type. One of the most comprehensive reference data models is the SAP application reference model, which includes more than 4,000 entity types. More than 180 business objects further cluster this data model (Curran and Keller, 1998). Further rules define relatively precisely the position of an entity type in this comprehensive model. A sophisticated navigation tool in SAP R/3 supports the use of the model. The modelling technique used is called SAP-SERM as it is a more structured language than the classical ER-data model. Independent entity types can be found on the left side of the model. The more an entity type depends on other entity types, the more it will be found on the right side of the model.

[3] Commonly these models are called 'data models' so we follow this terminology, although it would be more precise to call such a model a 'reference conceptual schema.'

Reference data models are particularly important for configuration decisions about the system organizational units as they depict the given opportunities of the System. A subset of the reference data model (approx. 30-40 entity types) allows a complete description of the inter-relations between system organizational units such as company, factory, distribution channel or division. This facilitates especially configuration decisions that cover more than one module. The main clusters of the data model are organized based on the functional modules and sub-modules of SAP.

17.4 Some Limitations of Existing Application Reference Models

Application reference models contribute significantly to the understandability of software functionality. However, they still have core weaknesses (Rosemann 2000), such as:

- As the models are focused on the description of the process execution and the information managed by them, it is not obvious what configuration alternatives exist. The analysis of a reference model shows what is possible in general, but not what might be recommended alternatives. Application reference models represent the entire functionality with the assumption that the complete functionality of the given application will be used: these models are not designed for configuration. Their modelling techniques do not support constructs that cover possible decisions during the implementation phase, *i.e.* decisions at build-time. Thus, they do not differentiate between decisions on instance level (in the particular implementation) and type level (in the reference data model);
- Application reference models concentrate on the elements that are of importance for a specific Enterprise System. Particular aspects of the organization, business objectives or manual tasks cannot be seen in these models. They do not include any references to the involved or required knowledge;
- Beside the missing transparency regarding possible choices during the configuration process, it is also not clear what consequences a configuration of one process or data structure has on other processes or data structures;
- Moreover, the models do not have any link to the actual process execution or database design. Thus, it is not possible (e.g. in the form of model attributes) to see the process performance expressed in key performance indicators such as processing time or resource utilization.

17.5 Configuration of Application Reference Models

17.5.1 Motivation

Application reference models are a comprehensive and consolidated description of possible processes and data structures within an off-the-shelf-solution.

This section is discussing how these models can be extended in order to depict configuration opportunities. The assumption is that this will increase the value of these models and their acceptance.

One criterion to distinguish different forms of reference models is their readiness to execute. This criterion characterizes the efforts that are required to convert a reference model into an enterprise-specific model. Two main forms of reference models can be distinguished in this regards. On the one side, a reference model can be 1:1 translated into an enterprise-specific model. Though it still might require some extensions (*e.g.* adding organizational units to a process model), it is in principle immediately a possible solution for a company. On the other side, a reference model may require configuration before it can be used. A model configuration step might be necessary for two main reasons. First, the model might depict exclusive alternatives (*e.g.* cost and accrual accounting). In these cases it is expected that the model user selects one of these alternatives. This process is similar to the required configuration of comprehensive off-the-shelf-packages that cannot be used without prior configuration. Second, a model might require configuration because it supports multiple purposes. In this case, the user has to select the purpose. Most of the currently available reference models are ready-to-run models and do not require any configuration. The only decision that has to be made is the identification of the relevant parts (*e.g.* procurement, but not human resource management). This simplifies and streamlines the use of these models as a complex configuration step requires related decisions, time and knowledge.

A more pragmatic explanation for the dominance of non-configurable models is that most reference models use classical modelling techniques such as Entity-Relationship Models, Class Diagrams, Petri Nets or Event-driven Process Chains. These techniques, however, are designed for individual enterprise models and not for reference models. The lack of dedicated languages for the design of reference models is a main reason why most models cannot be configured. Alternatives may only be depicted in the reference process model through redundant scenarios. The consequences of the limited capability to individualize a reference model are significant. Limited acceptance of the model can be expected, as it does not reflect the actual implementation. Thus, the model is not perceived as correct. Its relevance is questioned as it includes constructs without relevance. This all contributes to a restricted clarity of the model and a lack of adaptability.

The following two sections will continue using the SAP reference model as an example and discuss how the reference process model and the reference data model can be configured so that more detailed building blocks can be derived.

17.5.2 Configuration of Application Reference Process Models

Available application reference models are mostly focused on the description of the *execution* of processes. This is, for example, important for the documentation of the new processes for system users. The project team, however,

is interested in a process-oriented description of the possible *configurations* of the ERP processes. This would support the discussion of alternative process scenarios and further integrate the ERP implementation into the process improvement project.

For the purpose of describing the inherent customising opportunities and constraints, it is suggested to further extend existing application reference modelling techniques (Rosemann, 2000a). The suggestions are twofold. First, a description is provided of how reference process models could be enriched with further symbols so that the customising potential becomes transparent. Second, it will be shown how process models can be linked in order to highlight dependencies between processes. In both cases, EPCs will be used as an example.

The objective of this extended modelling technique is to describe different alternatives in one reference model. The optional system organizational unit "Business Area" within SAP's financial accounting solution (SAP-FI) is used as an example. Business Areas in SAP R/3 are defined as "the organizational unit in external accounting that corresponds to a selected area of activity or responsibility within an organization to which the value movements entered in financial accounting can be assigned." (SAP online documentation).

Although Business Areas are defined in SAP-FI, they are represented as a posting object and a reporting unit as part of most SAP modules. Consequently, the decision about the organizational unit Business Area influences many processes in several areas of SAP R/3. This influence, however, is not shown in the SAP reference process models.

Fig. 17.4(left side) includes a model for the relevant configuration process, in which the decision about the use of Business Areas is made. This process is strictly sequential as long as mandatory organizational units are configured (*e.g.*, the number and names of the legal entities, SAP term: Company Code). Decisions about optional organizational units are depicted as "check functions". After the decision has been made (*e.g.* "Business Areas are not relevant"), the configuration process for the organizational units takes place automatically.

A process which depends on the decision about the Business Area is the process of entering a new cost centre. If the Business Area is active, every cost centre has to refer to a Business Area. Thus, the configured model either includes the assignment of a cost centre to a Business Area or not. The reference process model "Entering a Cost Centre", however, has to depict both possibilities. Consequently, a special new connector is required, which describes this alternative. The XOR connector (circled with double line in the figure) is introduced for these purposes (Fig. 17.4, model in the middle). This connector indicates that an exclusive decision between the two alternatives has to be made. The classical XOR-connector on the other hand can only specify a runtime decision, this allows highlighting a decision at build-time. The connector includes a reference to the configuration process in which the decision has to be made, and the configuration process model is linked to the

operational processes that depend on customising decisions (Fig. 17.4, model on the right side). Thus, it is possible to clearly identify the influence of a particular customising decision. This example shows how reference process models could be extended in order to include more information about actual customising possibilities and process interdependencies. This facilitates the linkage between reference models and particular models.

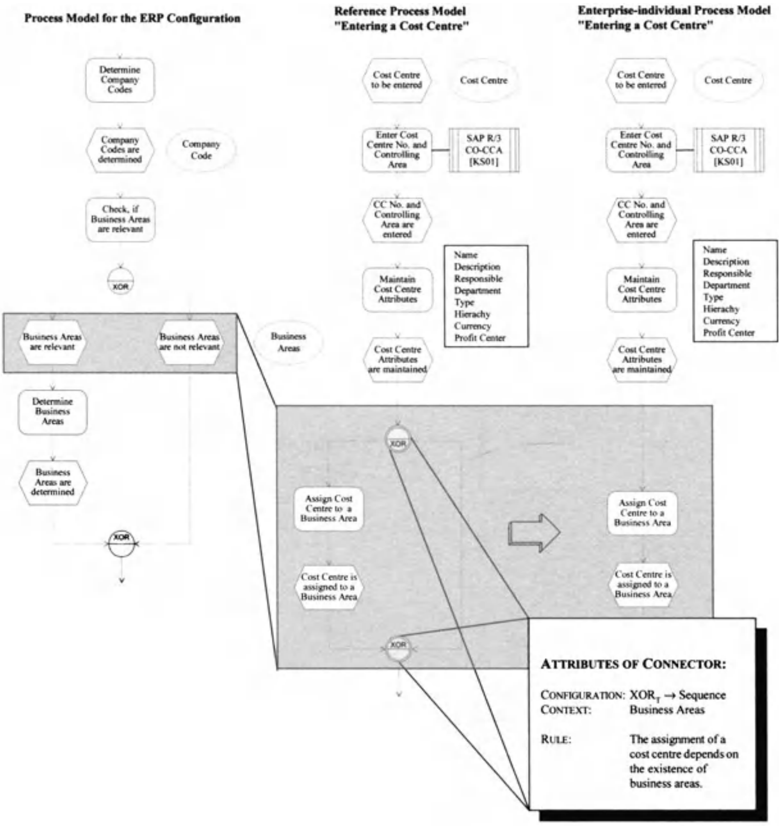


Fig. 17.4. Example of extended reference process models including inter-dependen-
cies between processes

17.5.3 Configuration of Application Reference Data Models

The following paragraphs discuss the main configuration decisions that can be made and how their possible representations in application reference data

models (Rosemann and Shanks, 2001). Extracts of the SAP reference data model are used as an example.

17.5.3.1 Optional Entity Types

Transparent examples for model configurations related to optional entity types can be found in ERP Systems in the definition of the organizational structure. The Financial Accounting solution in SAP R/3, for example, requires as discussed in the previous section a decision about the 'Business Area'.
In a reference data model, optional entity types such as a Business Area could be highlighted with a dotted line. Figure 17.5 shows the two entity types Business Area and Cost Centre. On the left side in Fig. 17.5 it is indicated that in the extract of the original SAP reference model a Cost Centre can refer to a Business Area. However, it is not clear if this can be configured. The proposed dotted line indicates that a Business Area is not necessarily required. The decision about this organizational unit is compulsory as other processes depend on this decision (see Section 17.5.2).

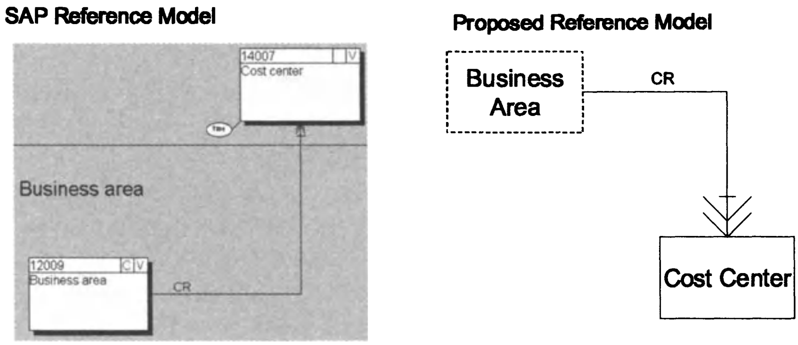


Fig. 17.5. Optional entity types with a required decision

Other important examples for such optional entity types are specializations of organizational units. During the system configuration process, the relevant specializations of an organizational unit have to be selected. Other organizational units are optional and the entire configuration process does not necessarily require a decision about these constructs. In these cases, the system typically sets one instance of this organizational unit as a hidden default. If a decision is made to use many instances of this organizational unit, an entity type in the corresponding data model is required. An example in SAP R/3 is the Dunning Area. A Dunning Area groups in Financial Accounting-Accounts Payable all customers that are treated in the same way when in comes to sending out reminder notices. No other process outside the dunning sub-module

depends on this decision. All customers are treated identically, if Dunning Areas are not used. The SAP reference model (Fig. 17.6, left side) only highlights that a Dunning Area existentially depends on a Company Code. In order to highlight the configuration potential, it is proposed to highlight optional entity types that do not require a decision during the system individualization with two dotted lines (Fig. 17.6, right side).

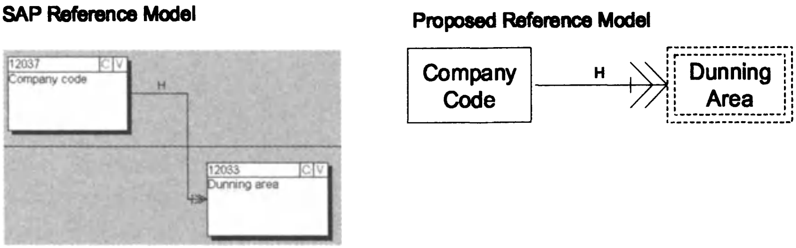


Fig. 17.6. Optional entity types with an optional decision.

17.5.3.2 Configuration of Relationship Types

The configuration of relationship types includes two decisions. Firstly, whether the relationship type is required at all, and secondly, what cardinalities should the relationship have. Optional relationships in ERP systems support a particular design alternative of cross-references. The more cross-references exist the more intensively different modules and sub-modules are linked. On the other hand, intensive cross-referencing demands a good understanding of these interdependencies from the system users and can be perceived as restrictive in daily processes.

17.5.3.3 Optional Relationship Types

Consequently, examples for optional relationship types can be found in the accounting modules of ERP systems. An example is given on the left side in Fig. 17.7, which shows the inter-relationship between Profit Centre and Cost Centre in SAP R/3. The arrow and the 'CR' (conditional-referential) indicates that a 0,m-0,1 relationship exists, *i.e.* each Cost Centre can refer to one Profit Centre. Each Profit Centre consolidates the cost of zero to many Cost Centres. However, it does not become clear, if the decision to link a Cost Centre to a Profit Centre can be made at build-time for all Cost Centres or only at runtime for each new Cost Centre separately. The proposed model (Fig. 17.7, right side) indicates clearly that this is a build-time (configuration) decision, *i.e.* the entire relationship type does not have to exist. The dotted line indicates that this is an optional relationship type.

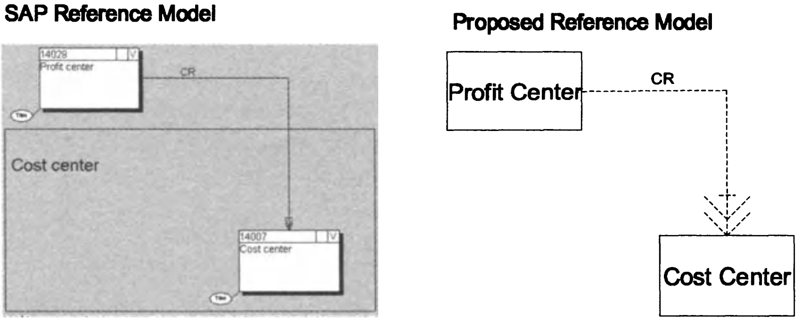


Fig. 17.7. Optional relationship type.

17.5.3.4 Configuration of Cardinalities

Decisions about the configuration of cardinalities can take place in conjunction with optional relationship types or independent from them. An example related to Fig. 17.5 would be that after establishing that the relationship type exists, a decision has to be made whether a Cost Centre has to necessarily refer to a Profit Centre or not. This difference can be expressed by the minimum cardinalities 0 and 1 and expresses a possible decision at runtime. In general, alternatives in ERP system configuration can be related to minimum and maximum cardinalities. Optional cardinalities that do not have to be defined during the customizing process should have a default.
Regarding the minimum cardinalities, a decision has to be made between 0 and 1, whereas 0 indicates that at runtime an entity of the involved entity type does not have to take part in the relationship. The typical alternatives for the maximum cardinality are 1 and many. An example for such a configuration alternative is given in Fig. 17.8. This example from SAP R/3 shows the inter-relation between Company Code, the highest reporting unit in Financial Accounting, and Controlling Area, the highest reporting unit in Cost Management (Fig. 17.8, left side). A variable is proposed to represent the maximum cardinality and to express that a Controlling Area either corresponds with exactly one Company Code ($x = 1$) or it covers more than one Company Code ($x =$ many). This is a mandatory decision during the configuration of SAP's accounting solution (see attached screenshot from the SAP configuration of the Controlling Area). Again, the existing SAP data model only includes the maximum case, *i.e.* $x =$ many. The actual configuration opportunity does not become obvious.

17.5.3.5 Inter-relationships Between Configurations

All the configurations presented above were local customizing decisions, *i.e.* each configuration could be made in the local context of the involved entity

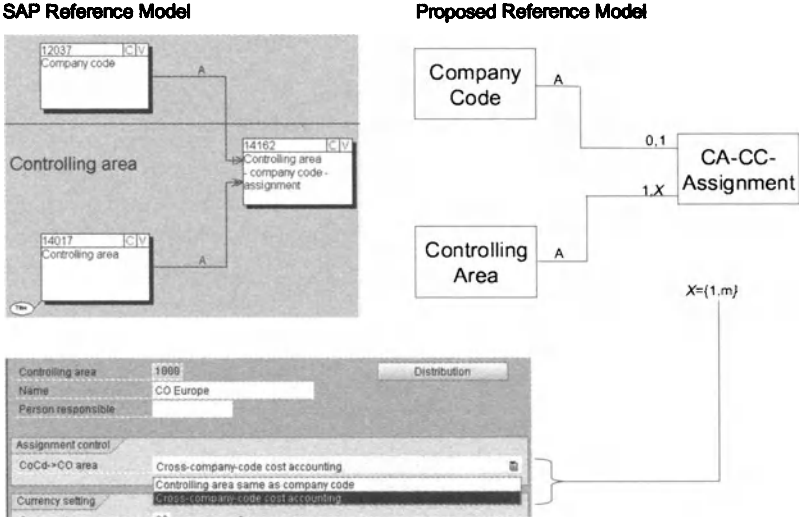


Fig. 17.8. Configuration of cardinalities.

types, relationship types and cardinalities. Further complexity is added, if inter-related configurations are also analysed. They require a combination of the suggested modifications of existing reference modelling techniques. Pointers would be useful in these cases in order to indicate existing dependencies. Another important case of configuration is re-configuration, *i.e.* when an existing system needs to be extended, *e.g.* after a period of system operation the configuration parameters may need to be changed. This of course has consequences to the detailed database design as follows: a) some changes can readily be adopted[4] , while b) in other cases the re-configuration decision may require a database re-load as part of the re-build process[5] .

17.6 Conclusion and Outlook

Reference models have been defined in this Chapter as reusable conceptual models that depict recommended structures and processes for organizations. One main class of reference models are application reference models that document in a semi-formal language the functionality of off-the-shelf-solutions. The SAP reference model has been presented as one comprehensive example

[4] *e.g.* addition of an optional attribute that had previously not been selected for use;

[5] *e.g.*changing a (0,1) (min-max) participation/cardinality constraint of a relation to a (0,n) constraint may change the way the relation is represented in the implementation - *i.e.*, change from representing the relation as an attribute (column) in a relational database to representing it as a separate table.

for an application reference model. Enterprise Modelling Languages used for representing Reference Models face specific requirements regarding the individualisation (specialisation) of these models. However, current models such as the SAP reference models (and other ERP systems' reference models) are mostly designed using modelling languages that do not have provision for the representation of various model dependencies in general, and model individualisation in particular. Thus, only limited opportunities exist to specify alternative scenarios. This Chapter has proposed extensions of popular modelling languages that allow exactly this specification of explicit configuration potential in reference process models and reference data models.
Future challenges for reference models will be the satisfaction of multiple purposes and an increased variety of model users. Reference models will have to take into account the context in which they are used. The growing popularity of business modelling in addition to software development and the resulting involvement of business users without any experiences in conceptual modelling are main developments. For example, it will be required to include in the reference models relevant benchmarking information or typical configuration decisions made in one industry sector.

References

Becker, J. and Schütte, R.(1996). Handelsinformationssysteme (in German, Merchandising Information Systems). Verlag Moderne Industrie: Landsberg/Lech.

Bernus, P. and Nemes, L. (1996).A Framework to Define a Generic Enterprise Reference Architecture and Methodology, Computer Integrated Manufacturing Systems. Vol. 9, No. 3, pp.179-191

ISO/TC184/SC5/WG1. (1999). Annex A: GERAM. In ISO/DIS 15704: Industrial automation systems - Requirements for enterprise-reference architectures and methodologies. Also: IFIP/IFAC Task Force on Enterprise Integration. GERAM V1.6.3 (available from http://www.cit.gu.edu.au/~bernus)

CCTA (2000). ITIL Service Support. The Stationery Office Books.

CCTA (2001). ITIL Service Delivery. The Stationery Office Agencies.

Chen, P. (1976). Entity-Relationship Model: Towards a Unified View of Data, ACM Transactions on Database Systems. Vol. 1, No. 1, pp. 9-36.

Curran, T. and Keller, G. (1998). SAP R/3 Business Blueprint. Prentice Hall: Upper Saddle River.

Davis, R. (2001). Business Process Modelling with ARIS: A Practical Guide. Springer: Berlin et al.

Green, P. and Rosemann, M. (2000). Integrated Process Modelling: An Ontological Evaluation. Information Systems, Vol. 25, No. 2, 2000, pp. 73-87.

Gulla, J.A. and Brasethvik, T. (2000). On the Challenges of Business Modelling in Large Scale Reengineering Projects, Proceedings of the 4^{th} Inter-

national Conference on Requirements Engineering. Schaumburg, Ill., 19-23 June, pp. 17-26.

Keller, G. and Teufel, Th. (1998). SAP R/3 Process Oriented Implementation: Iterative Process Prototyping. Addison-Wesley.

Kirchmer, M. (2000). Business Process oriented Implementation of Standard Software. 2^{nd} ed., Springer-Verlag: Berlin, Heidelberg.

Rosemann, M. (2000a). Using Reference Models within the Enterprise Resource Planning Lifecycle. Australian Accounting Review, Vol. 10, No. 3, November, pp. 19-30.

Rosemann, M. and Green, P. (2000b). Integrating Multi-Perspectives into Ontologies. Proceedings of the 21^{st} International Conference on Information Systems (ICIS 2000) (CD-ROM). Eds.: W. J. Orlikowski et al., Brisbane, 10-13 December 2000.

Rosemann, M. and Shanks, G. (2001). Extension and Configuration of Reference Models for Enterprise Resource Planning Systems. Proceedings of the 12^{th} Australasian Conference on Information Systems (ACIS 2001), eds.: G. Finnie, D. Cecez-Kecmanovic, B. Lo. Coffs Harbour, 4-7 December, pp. 537-546.

Scheer, A.-W. (1998). Business Process Engineering. Reference Models for Industrial Enterprises. Springer: Berlin.

Scheer, A.-W. (2000). ARIS. Business Process Modelling. 3^{rd} ed., Springer: Berlin.

Silverston, L. (2001a). The Data Model Resource Book, Volume 1, A Library of Universal Data Models for All Enterprises, Revised Edition. John Wiley & Sons.

Silverston, L. (2001b). The Data Model Resource Book, Volume 2, A Library of Data Models for Specific Industries, Revised Edition. John Wiley & Sons.

TelemanagementForum (2001). eTOM. The Business Process Framework for the Information and Communications Services Industry. Public Evaluation Version 2.5. Guidebook Number 921. December.

Verbeek, M. (1998). On Tools & Models. In: Dynamic Enterprise Innovation. Establishing Continuous Improvement in Business. 3^{rd} ed., Ed.: R. van Es. Baan Business Innovation.

18

DESIGNING THE INFORMATION TECHNOLOGY SUBSYSTEM*

Luis M. Camarinha-Matos[1] and Hamideh Afsarmanesh[2]

[1] New University of Lisbon, Faculty of Sciences and Technology, Portugal
[2] University of Amsterdam, Faculty of Sciences, The Netherlands

18.1 Introduction

The Information Technology (IT) subsystem as a component and enabler of the business' information system (IS), is a critical core element of the infrastructure of any modern enterprise. In fact, from the users' perspective, the software side of the IT is represented by a large variety of tools, ranging from the generic office tools to specialized engineering, production planning, management decision support tools, or even large applications covering a wide spectrum of functions, such as the ERP systems. In order to guarantee the interoperation among these tools, or at least the necessary level of information exchange, a proper network infrastructure must also be in place. When the interoperation / integration among different IT components of the enterprise is ensured and their activities are coordinated, the IS, and implicitly IT subsystems play an essential, supporting role in the company's performance and efficiency.

How to design an enterprise IT subsystem?

From a methodological perspective, a first approach would be the use of the typical top-down design and analysis method from traditional software engineering. This approach however, is only appropriate when the task is to design a new system from scratch.
Considering the large 'legacy' software components that most companies have, or that is still available on the market, the problem is more often a matter of component integration, i.e. how to glue together the various existing software components and build / source the missing parts, so that a seamless operation can be achieved. In this context a bottom-up (or at least hybrid) approach to analysis and design is more adequate.

* This work is supported in part by the European Commission through the IST THINKcreative and VOSTER projects.

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

When starting with a high-level architecture design of a desired system, based on its specific requirements, it is typically necessary to analyze its individual legacy components and devise methods to properly integrate them. This is a critical task for which there are no simple solutions and that very often requires customized solutions on a case-by-case basis.
The following sections aim to achieve a more detailed description of the integration challenge and to present to some possible practical approaches and methodologies, rather than attempting to find a universal solution. The need for integration and its facets, levels and obstacles are first discussed, followed by a set of general steps for systems integration. Systems integration is then analyzed at various levels of complexity[3]. The main applied technologies, typical solutions, main trends, and open challenges are subsequently further discussed and exemplified. In the following, manufacturing is taken as the example sector to illustrate the addressed subjects.

18.2 Integration Levels

18.2.1 The Need for Systems Integration

What is systems integration?
The issue of systems integration has been consistently on the research and development agenda (especially since the 1980s) for the manufacturing enterprise systems (Camarinha-Matos et al. , 1997). But what is integration? Common meanings of integration are a) a process through which individuals of a lower order get together to form individuals of a higher order and b) to make the subject of integration a whole, or to complete it.
Specifically, in the manufacturing domain we can consider that integration is the process through which a set of components such as machine tools, robots, sensors, feeders, conveyors, application programs, people, etc (generally referred to as *resources*) unite to form compound resources or systems / sub-systems of a higher order.
This type of union comprises a notion of complementarity and a "fluid" interoperation among the components. In other words, it implies the creation of proper conditions for various components (independently of their level of autonomy) to be able to dialog and cooperate in order to achieve the goals of the manufacturing system. A manufacturing system may be regarded as a set of physical and logical entities organized for the fulfillment of some objectives (e.g. production of physical or virtual products). In this view, the integration process aims to optimize the co-operation among these entities.
An example of integration process is the creation of a manufacturing cell out of various components sourced from different suppliers (Fig. 18.1).

[3] for instance, the three-level integration (shop-floor, intra-enterprise and inter-enterprise) integration common in manufacturing

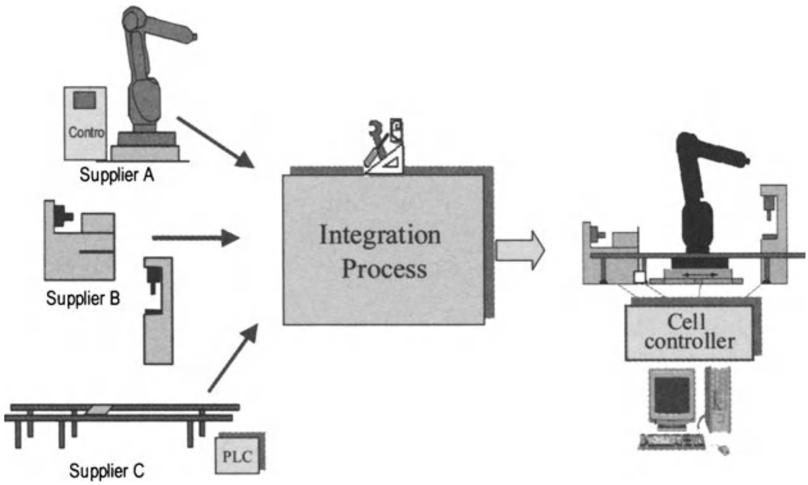


Fig. 18.1. Integration of components in a manufacturing cell

Why systems integration?

Systems integration, embedding the interoperation among components / subsystems and their coordination, is an important requirement in order to support the following important facets of organizations:

- Flexibility – integration may support the capability of a system to rapidly adapt to various necessary processes[4].
- Agility – integration may enhance the capability of a system to rapidly adjust to unforeseen changes, both in the external and internal environments. The value of an enterprise (and its sub-systems) is increasingly determined by its capability to change.
- Efficiency – may be boosted by avoiding unnecessary activities, for instance the generation of information that already exists in the systems and that can readily flow from one system to another. By allowing information to better reach all areas of the company / organization, systems integration provides the basis for a higher co-responsibility and participation of all actors in the organization.
- Quality – the automation of information exchange processes brought about by integration increases the quality by reducing the number of potential errors caused by manual information input. Furthermore, the achieved seamless flow of information and control contributes to an increased quality in terms of response time and supports an informed decision-making process by providing updated information.

[4] For instance, manufacturing of several product variants is an important property when mass customization is growing in importance.

These are some of the most important reasons for systems integration for companies facing high competition and rapid market changes. However, systems integration, by providing a global view of the complete system at the enterprise, can also tackle other new organizational aspects. Other indirect benefits can be gained, such as:

- Preservation of knowledge about the entire system;
- Higher robustness against situations of local inefficiency;
- More optimized supervision / coordination / management of activities;
- Support for cost estimation;
- Possibility of global planning / scheduling of activities.

It is important to note the substantial emphasis laid nowadays by the companies on approaches such as knowledge acquisition, knowledge organization, knowledge management, and knowledge sharing. Systems integration facilitates the organization of the knowledge about the system.
It is also important to note that the recent developments in the information and communication technologies have paved the way for the fulfillment of the increasing efforts put into systems integration during last decades. The recent advances in the IT technologies have created new possibilities and enablers for systems integration.

18.2.2 Facets and Levels of Integration

Considering the multi-faceted characteristics of each subsystem, systems integration must be seen from different **perspectives** according to the various focused facets. Some of the main perspectives of integration are:

- Functional – in which the objective is the establishment of a coherent set of complementary functions / activities, and their coordinated processes, in order to achieve the global system's objectives.
- Information - in this perspective each component / sub-system may produce and consume information. The harmonization of information models / building of common abstractions (from which partial views may be derived for each component) naming, semantics and the elimination of redundancies and inconsistencies are some of the aspects of information integration.
- Interoperability mechanisms and communication – under this perspective the focus is put on the communication needs for interaction / interoperation among components / sub-systems. The definition of protocols and common interfacing mechanisms are some of the aspects of systems integration. The coupling forms (tight, loose, distributed), the physical channels (networks) and the human-machine interfaces are also part of this perspective.
- Coordination – the effective use of an integrated set of entities depends on the adequate orchestration (coordination) of their "participation" in the

processes running in the system. The establishment of a global coordination structure is therefore a requirement for systems integration.

Furthermore, the systems integration need to be addressed and instantiated at different levels of complexity and abstraction (Fig. 18.2), as follows:

- Cell level – when basic resources (robots, NC machines, conveyors, etc.) and their local controllers need to be integrated in order to build a cell dedicated to a specific function or a set of functions (assembly, painting, inspection, etc.);
- Shop-floor level – when various cells, transportation subsystems and warehouses are integrated within one a manufacturing system;
- Intra-enterprise level – when the objective is to integrate all areas of the enterprise, including not only the shop-floor but also other departments e.g. marketing, planning, engineering, etc. and their interactions;
- Inter-enterprise level – when cooperation among various enterprises is envisaged. The manufacturing processes or complex services are not performed by isolated companies. On the contrary, in a network of enterprises (virtual enterprise) each node contributes with some value to the value chain. The materialization of this paradigm requires the definition of a reference architecture for the cooperation process and the development of a support infrastructure, including the protocols and services for information exchange, communication and cooperation.

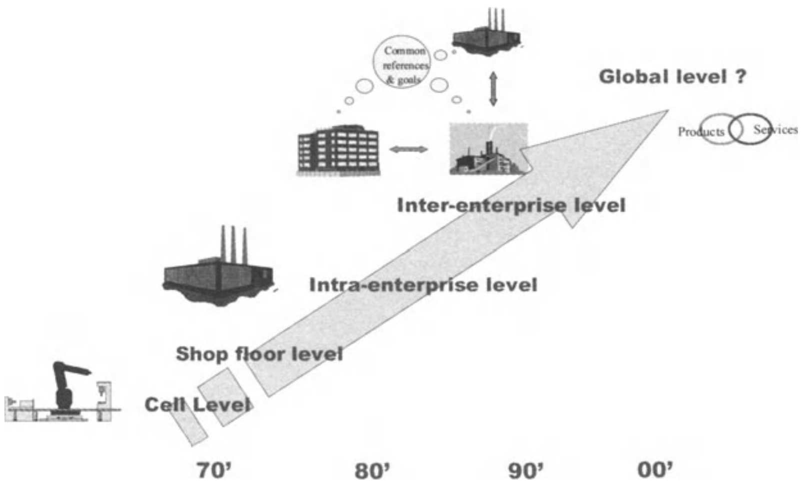


Fig. 18.2. - Levels of integration in manufacturing enterprises

Furthermore, the need for a new level of integration (integration at global level) is emerging as a result of propagation of the so-called *ubiquitous* or *pervasive* computing. The inclusion of processing capabilities (local intelligence)

is spreading over all surrounding environments, both in the professional environment or at home (Fig. 18.3). The working methods change, enabling to perform the professional activities from different locations[5] .

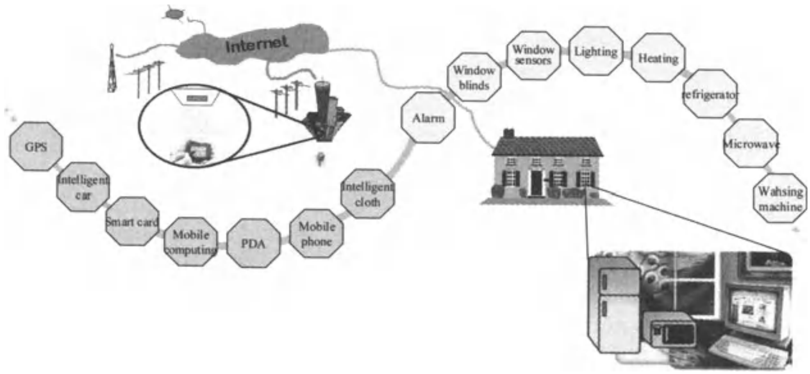


Fig. 18.3. Ubiquitous computing and global integration

This tendency is reflected by the proliferation of intelligent devices such as Personal Digital Assistants (PDAs), mobile phones, smart cards, embedded networks in vehicles, processors embedded in the athletes' or patients' garments (in order to monitor their status), elevators, safety and surveillance systems, traffic control systems, intelligent and Internet-enabled home appliances, etc. An important challenge is the interoperability among all these components and the development of appropriate integration approaches[6].

18.2.3 Obstacles to Integration

Systems integration is in general, a complex process facing a number of obstacles, such as:

- Heterogeneity. The components to be integrated might be highly heterogeneous due to:
  - *Abstraction levels* – different components may operate or be operated at different abstraction levels.
  - *Dimension / scope* - there might be a need to integrate components of different dimension and diversified operating scope.
  - *Supporting technologies* - the implementation and support technologies (e.g. programming languages, information management systems, etc.)

[5] the so-called *telecommuting*.

[6] a problem that emerged early in the dawn of automation are the so-called *islands of automation*, whereby automated clusters cannot 'talk' to each other.

used in the various components may belong to different technologic branches and generations.
  - *Phases of the life cycle* – the various components being integrated (new and legacy systems) may have different life cycles and be in different stages of their life histories .
  - *Interfacing approaches and languages*– different components offer (implement) different mechanisms to interface to external systems.
  - *Underlying concepts and modeling formalisms* – the underlying paradigms as well as the ontologies and modeling formalism adopted by different components are diverse.
- Distribution. The components to be integrated may have a physical / geographical distribution (e.g. located in different locations of the shop floor, in different sites of the same company, or even in different companies) or a logical distribution (having their computational parts installed on different computers and/or software applications).
- Autonomy. Most legacy systems were not designed to be integrated within a global system. Therefore, from a control perspective, they possess a high degree of autonomy but are not very "sociable". This is, for instance, the case with many robot controllers that were designed as if the robot was the "center of the universe". An important challenge is therefore to overcome (at least in part) the control structures embedded in the legacy systems while recovering their functionality or services[7].
- Continuous and rapid technologic evolution. The information and communication technologies are facing a very fast evolution, leading to products with very short life cycles, what implies a continuous substitutions (re-integration) of new versions of the products within the old versions. Such evolution contributes to the increase of the heterogeneity problems and requires integration to become a continuous process, coping with different life cycles and different phases of the life cycle.
- Multi-disciplinarity. An enterprise or manufacturing system involves many areas, disciplines, and technologies. Consequently, the systems integrator is required to have the knowledge to on one hand understand and handle a large variety of topics and contexts and on the other hand to tackle new areas that are continuously emerging as a result of the technologic evolution.

In this context the Information and Communication (ICT) infrastructure is frequently aimed at playing an *intermediary* role as an enabler of the inter-operation among components. Ideally in the future the infrastructure should support "plug and play" of new components in a seamless way. Furthermore, and from another perspective, the enterprise integrating infrastructure should play the role of "*enterprise operating system*" or executor, hiding the details

[7] this may involve reverse-engineering their control software (since many legacy software systems have been built without adequate documentation).

of the machinery of its complex systems (a vision that goes back to the CIM-OSA (AMICE , 1993)).

18.2.4 Historic Overview and Technologies

Systems integration (under various names) has been a major topic of research and development for the last three decades.
According to Pinto (1999), the evolution in enterprise integration may be represented by three main stages (Fig. 18.4).

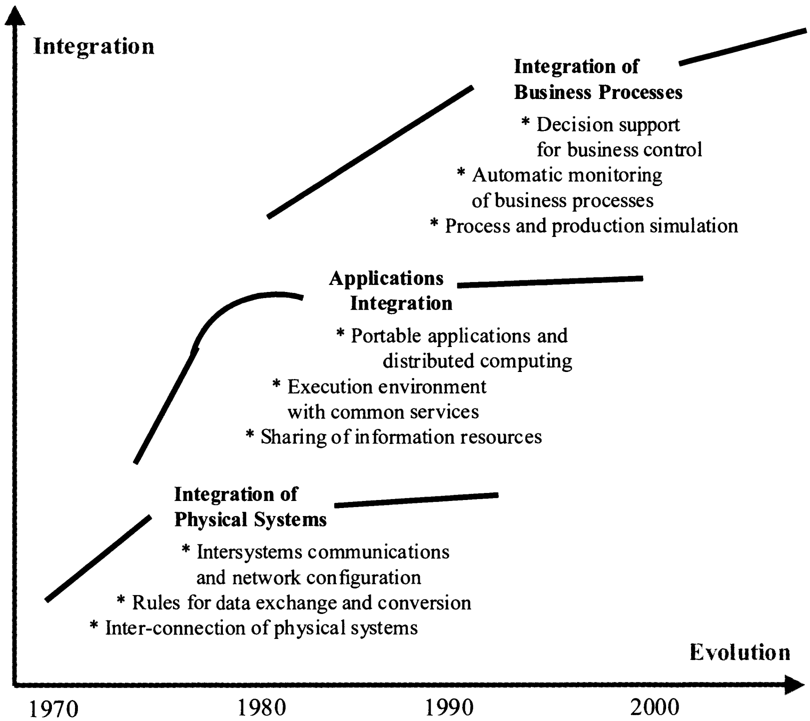


Fig. 18.4. Levels and phases of enterprise integration

However, Fig. 18.4 may lead to the wrong conclusion that nowadays the first two challenges of integration of physical systems and applications are resolved and no longer a serious problem.
Another vision of the "history" of manufacturing enterprise integration can be the one shown in Fig. 18.5, where in fact the integration work at the three levels of abstraction continues through the three decades.
This picture is not intended to be complete (i.e. showing all the paradigms and development areas in systems integration). Neither is it strictly accurate

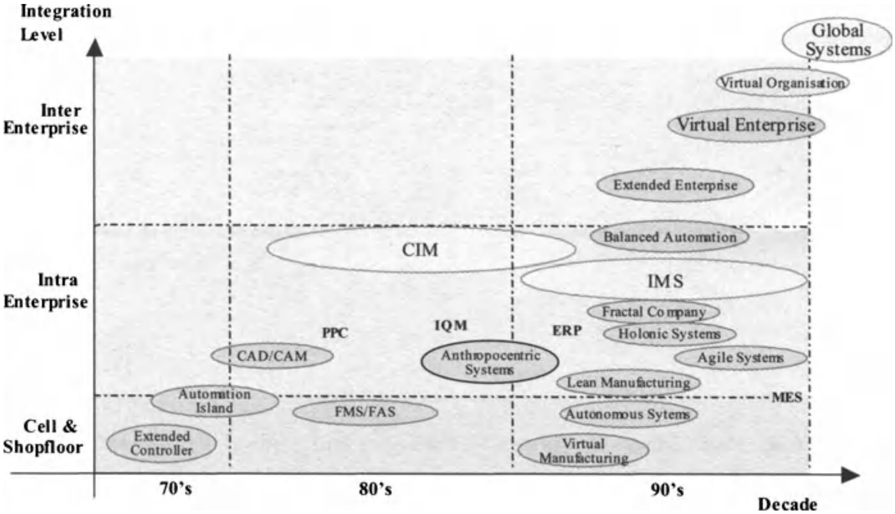


Fig. 18.5. Main phases in manufacturing systems integration

in terms of the precise time span for each paradigm. Rather, the purpose is to provide a general and simplified overview of relative relationships of the different integration developments. For instance, the ellipse representing CIM does not mean that this topic "finished" in the early 1990's, but that it received less attention since then and the relevant developments slowed down. Similarly, the idea is to show that the second half of the 1980's were the most active years for this paradigm.
Also, as it can be seen in the same figure, in recent years increasing attention is being devoted to the integration of more complex or global system, however the integration issues at the cell or shop-floor levels still remain on the agenda and not resolved.
The actual implementation tools used for systems integration depend on the technologies available during each historic phase both for components development and integration support. A very simplified overview of the main paradigms and technologies used in manufacturing systems integration during the last three decades is shown in Fig. 18.6.
It should be noted that the increase of systems complexity and integration scope is followed by an increase and diversity of the potential available paradigms and technologies. We are in fact facing a scenario of an excessive number of technologies suggested and produced by different developers, which corresponds to an excessive number of promises. In fact, each new paradigm and technology tends to promise the fulfillment of most capabilities of the other similar products while solving all problems of their previous generations, promise that in reality hardly materializes. On the contrary, the multiplication of all tools by the fast introduction of new versions and generations of those

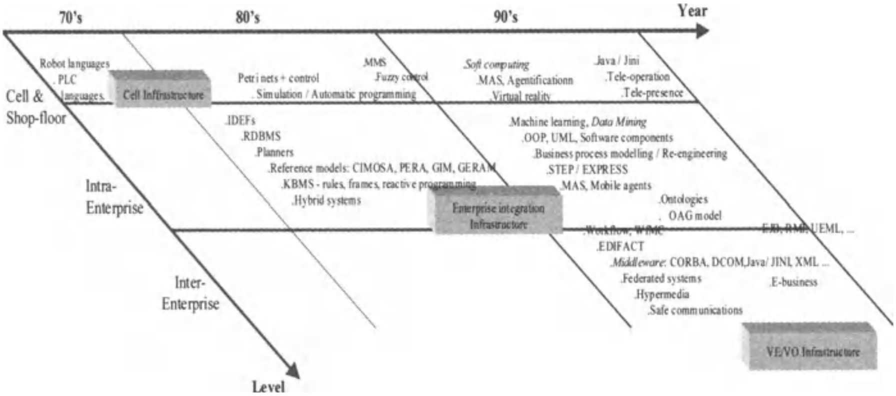


Fig. 18.6. Main integration technologies and support paradigms.

tools greatly increases the incompatibility ratio among components, which in turn justifies the question "To what extent are these technologies and tools enablers, or are they rather *disablers* of systems integration?"
In addition to the diversity of paradigms and technologies available at a given historic phase, there is also always a co-existence in each enterprise of diverse technology generations and components with different life cycles and in different phases of their life histories . Therefore, the systems integrator has to master not only the current tools of the current time frame, but has to also take into account the legacy systems and to promote their technological migration.

18.2.5 Towards a Methodology for Systems Integration

As the specific integration techniques to be used in each context evolve according to the various technology generations, and each integration problem shows some peculiarities due to the heterogeneity of involved components, it is difficult to establish general methodologies for systems integration. Nevertheless there are some **major steps in systems integration** that are relatively independent of the used technologies. The following steps represent a generic methodology to be used for systems integration:

- Use / adoption of reference models
  Similar to the process of assembling a puzzle, systems integration can be simpler (or less difficult) when guided by a reference model. For instance, at the *enterprise level integration* it is important to resort to a "guide" like GERAM (IFIP-IFAC , 1999). However, while solving a puzzle there is a guarantee that all little pieces have their own position / role in the global picture and there is exactly one piece for each position of the puzzle. Similar conditions cannot be fully guaranteed in systems integration.

Furthermore, in some cases there is no reference model defined yet that may be followed.

- Systems modelling
In addition to the use of generic reference models, there is a need to model the particular aspects of the target system, in order to allow its understanding and assessment, or the communication among members of the integration team. This modeling process shall consider different aspects: Functional (components and processes), Information, Resources, Organization, etc. When the objective is to improve an existing system (perhaps the most frequent case) there is a need to at least consider two models: model of the initial system (AS-IS) and model of the target system (WOULD-BE).

- Harmonization of components interactions
The heterogeneity of the components to be integrated requires an effort to adapt or harmonize them, namely in terms of the interaction mechanisms and the way the services and components can be accessed. Building individual *wrappers* around a legacy system (either directed to a pair-wise or to a common model of interaction) is a typical example supporting the harmonization. *Componentization* is another approach that can be applied in this step. On the other hand, some applications, particularly in new and not yet well structured areas, tend to grow in a monolithic fashion, which impedes their integration with other components that might have some functional overlaps. In these cases, it is necessary to apply a "fragmentation" process, i.e. identification and characterization of the basic functionalities involved and their separation into interoperable modules, following some kind of componentization process. It is implicit that this process requires the establishment of common packaging of components and interfacing communications infrastructure (and adequate protocols) in order to support interaction among components running in a distributed computational environment.

- Information / knowledge inter-linking
Complementarily to the interfacing harmonization, there is a frequent need to establish information / knowledge integration mappings in order to overcome the semantic incompatibility between the concepts used by different components. Establishment of common ontologies or adoption of information standards (e.g. STEP[8] in the case of technical product data) will reduce pair-wise mapping needs. However, there are many classes of information in enterprises of different sectors that are not standardized. Even when a component is supposed to be compliant with a given standard, some problems might arise due to the use of different versions or differ-

[8] Standard for the Exchange of Product Data (ISO1994)

ent sub-sets of the same standard. This is, for instance, a typical problem when adopting EDIFACT[9]. In most cases, still the creation of mappings to resolve the heterogeneity of information / knowledge semantics is unavoidable.

- New coordination levels
  Finally, once the interoperability among the various components is guaranteed, there is a need to establish a new system coordination, or a control layer. This layer is responsible to guarantee that the behavior of various components is orchestrated towards achievement of the global objectives of the integrated system. It is important to notice that part of the autonomy of each component has to be relinquished or transferred to this coordination layer. One objective when building a wrapper around a component is to allow the access to the services implemented within the legacy system while hiding the legacy control structure.
  The Integrating Infrastructure plays the role of "operating-" or "execution system" for the integrated system. Some of its main roles are the orchestration of components, task assignments and task execution, the management of dependencies (e.g. synchronization) and error handling.

18.2.6 Integrating Infrastructure

Design and development of infrastructures to support integration must provide not only the basic interoperation support mechanisms, but also a set of generic services to support cooperation among the sub-systems. In fact, there is a growing awareness of the need to address the problems of interoperability and cooperation / collaboration in a systematic way.
Fig. 18.7 illustrates some of the main topics in the current research and development agenda for the various levels of integration of industrial systems. A similar agenda can be inferred for other application sectors.
It is important to note that although the cell/shop-floor and enterprise levels integration have been the main focus of attention in the past three decades, they still remain on this agenda, with a somewhat different focus of attention. It should be noted that clearly, in the past developments not all steps of the integration methodology suggested above have received the same emphasis from practitioners in each of the three levels of integration. For instance, cell and shop-floor integration has traditionally focused more on the "harmonization of components interactions", in which Multiagent Automation Systems (MAS) tends to play a dominant role. At the same time, only lately the issue of "new coordination levels" started to be addressed, including design of intelligent supervision systems, flexible coordination, and shop-floor re-engineering.
One of the most addressed areas at the intra-enterprise level has been the "information / knowledge inter-linking" step, with particular relevance to the

[9] Electronic Data Interchange for Administration, Commerce and Transport (ISO 9735 and related standards)

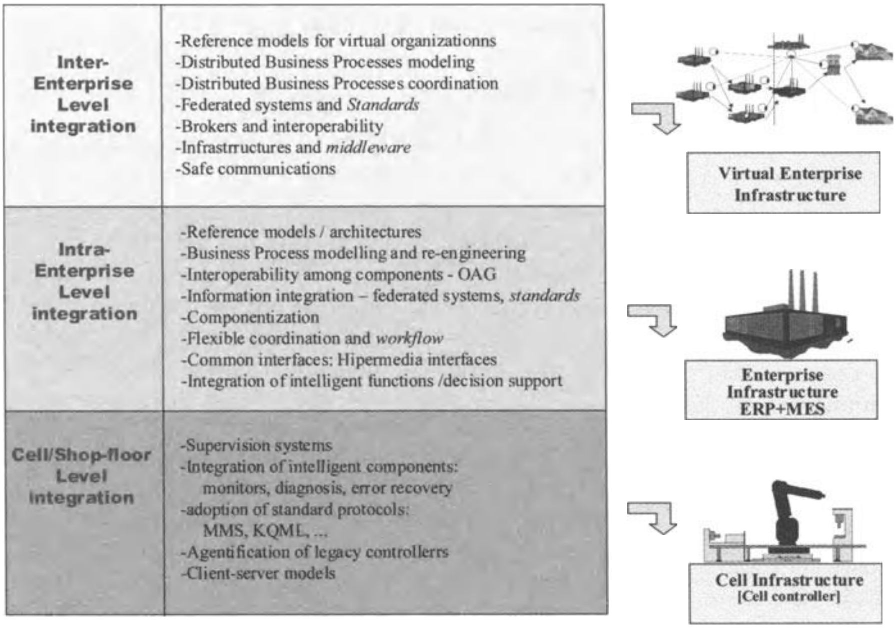


Fig. 18.7. Current agenda in industry systems integration

information integration efforts. But aspects such as “adoption of reference models”, “systems modeling” and “coordination levels” have also been addressed.

For the inter-enterprise level (being a more recent area) there are not yet common reference models, and most emphasis has been put on “harmonization of components interactions”, “information / knowledge inter-linking”, and distributed “coordination levels”. When collaborative networks, under different organizational forms, are becoming more pervasive, a new *entry* in the infrastructures design that may have substantial impact at the inter-enterprise cooperation level is *grid-like computing.*

Practical solutions in the mentioned areas have focused on specific aspects (perhaps the ones perceived as the first priority in the area) while several other steps remain as open challenges. Therefore, the next sections of this document, instead of exemplifying the multiple-step methodology for each level, rather present details of the most common technological approaches and adopted techniques.

18.3 Shop-floor Infrastructure Design

18.3.1 Brief Historic Overview

Manufacturing shop-floor is historically one of the first areas where substantial integration efforts were made and where the first "islands of automation" emerged[10]. These efforts were particularly strong during the 1980's and early 1990's, when many research projects in this area were funded worldwide.
Nevertheless, this area is still one of the most challenging and difficult endeavors in terms of systems integration. Heterogeneity is a general rule in manufacturing shop-floor components. Unlike other areas of the enterprise, there is no other option but to combine components originated from different manufacturers. Furthermore, the various components (and their controllers) show a large diversity in terms of the level of complexity. The typical duration of the life cycle of these components is substantially higher than other "purely software" components and, therefore, there is a need to cope with different technologic generations in the same environment. Finally, real-time constraints and the need to interface with physical devices (which are not always easy to model and whose behavior may evolve / degrade over time) add to the complexity of the task.
Early controllers (and their associated programming languages) were fundamentally "component-centric" and with almost no support for interaction with other components. That is typically the case with robot controllers – most of the early generations assumed the robot as the "center of the world" and, at most, would consider a very limited interaction with external devices and sensors, e.g. via binary signals.
Following the trends in Computer Science, new programming paradigms were slowly penetrating the automation world and controllers and their programming languages became more modular and started to consider the interaction with other complex sub-systems[11] , moving from a component-centric to a system-aware approach .
With the boom of the PCs and LANs witnessed in the 1990's, the concept of client-server entered the shop-floor at the same time with the object-orientation and more user-friendly real-time operating systems became popular.
Standardization, first at the component level and later at the systems level, has always been a subject of attention of the shop-floor automation community, but the progress has been very slow. In spite of some success in the area of low-level programming of NC (numeric control) machines, the standardization attempts have been, in general, quite unsuccessful, even at component level. That is the case of robot controllers and robot programming systems.

---

[10] e.g. the integration of closely related equipment controllers, namely through Programmable Logic Controllers (PLCs).

[11] e.g. interaction between a robot controller and a vision system.

Even when these efforts resulted from the initiative of consortia of large manufacturing suppliers, as it was the case of the Manufacturing Automation Protocol (MAP)/ Manufacturing Messages Subsystem (MMS) initiative, the practical impact was rather limited, making shop-floor integration a very hard and expensive task.
In a scenario where customized solutions have to be built on a case-by-case basis, middleware solutions based on Remote Procedure Call (RPC), Common Object Request Broker Architecture (CORBA), Distributed Component Object Model (DCOM), Java/Remote Method Invocation (RMI), or even Jini (Sun , 1999), together with the concept of components-based programming, became common tools in industrial practice.
The progressive move towards more integrated shop-floors and ultimately the establishment of a kind of shop-floor operating or execution system (that not only provides a high level view of the manufacturing system but also establishes the link with the other enterprise sub-systems), has led to the concept of Manufacturing Execution System (MES).

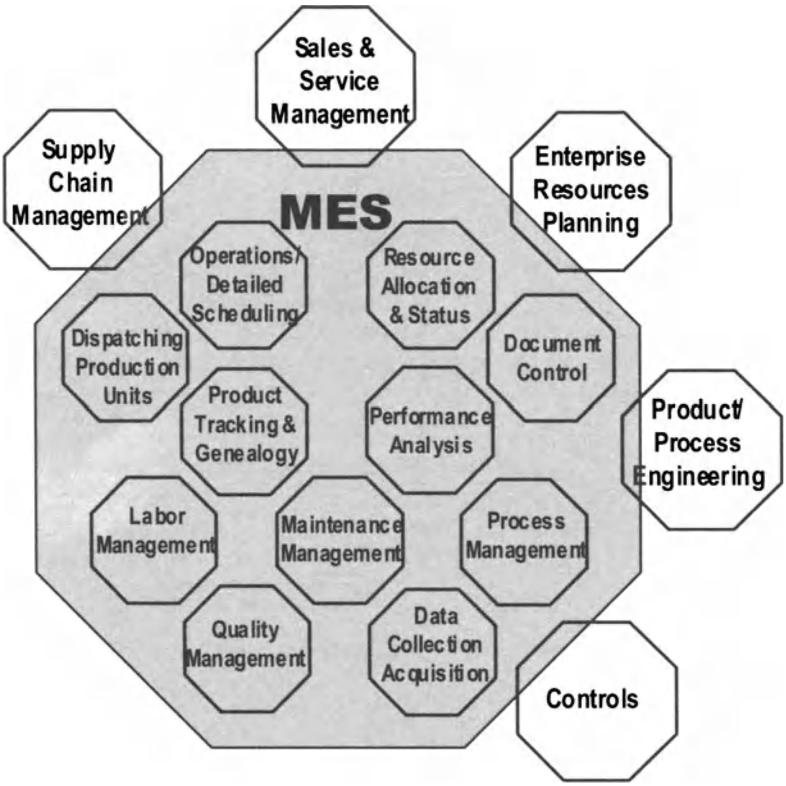


Fig. 18.8. Manufacturing Execution System functions.

A MES is typically understood as an integrating component that delivers information on the manufacturing processes and manufacturing resources status, enabling the optimization of production activities, from production orders launch to finished products. In this way, MES becomes of particular value to the operational management.

Figure 18.8 shows the position of MES with regard to other enterprise subsystems, according to the somewhat extended view of the Manufacturing Enterprise Solutions Association (International). As the figure illustrates, the MES system does not eliminate the problem of integration with legacy controllers (which remains to be resolved on a case-by-case basis), but rather represents a high level layer that acts on top of the various resource controllers.

In parallel with the integration trend, during the last decade considerable effort was also spent on the definition of intelligent supervision architectures that progressively included error detection and diagnosis, error recovery, and machine learning functionalities ((Lopes and Camarinha-Matos , 1999; Basseto et al., 2002)), as presented in Fig. 18.9. Since an effective supervision system requires both access to- and control over all the subsystems, their development must be intertwined with the systems integration trends.

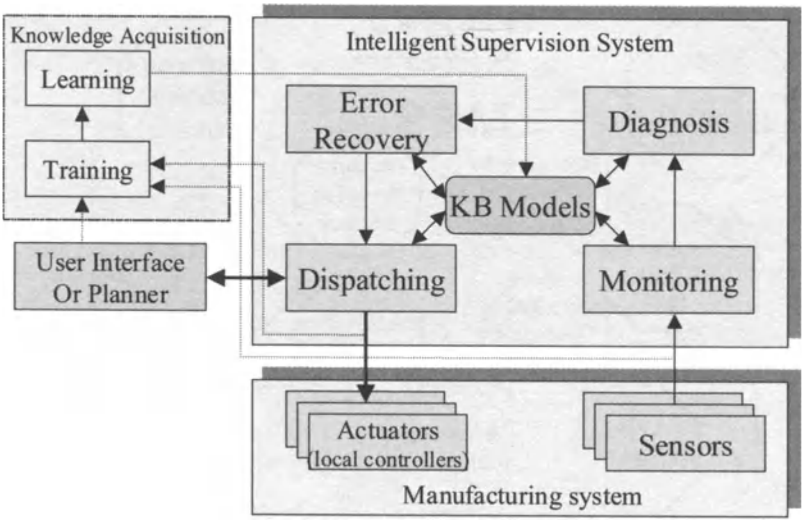


Fig. 18.9. Intelligent Supervision System architecture

Finally, it is important to highlight a recent and quite promising entry in the shop-floor integration arena: the Multi-Agent Systems (MAS) technology. MAS is expected to have a large impact on advanced shop-floor infrastructures and therefore justifies a particular attention in the following sections.

18.3.2 Integration of Legacy Controllers

Migration of legacy systems into a form that allows their integration represents a difficult task and requires an analysis on a case-by-case basis. Typical techniques that have been considered an applied for this purpose are addressed in this section:

- Construction of wrappers around the legacy systems. The goal is usually to transform the legacy controller into a server (under a client-server model) in such a way that the fundamental application services provided by the legacy system are recoverable, while its control structure and user interface are discarded (Camarinha-Matos et al. , 1997). Figure 18.10 illustrates the approach.

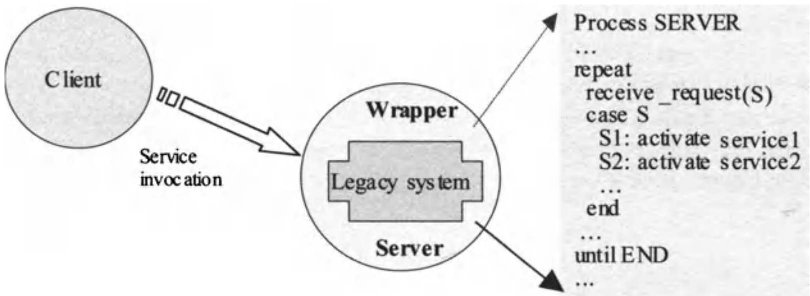


Fig. 18.10. Wrappers in legacy systems migration

The main difficulties in this case, for which no general rule can be drawn, are the separation between the services and the (monolithic) application control structure of the legacy system.
- Utilization of middleware layers that were designed to support the development of distributed applications, giving the client objects or components the possibility to invoke, in a transparent way, the methods provided by an object or component server running on the same or any other machine in the network. Examples of middleware layers are CORBA, DCOM, JINI, etc. If such a layer is not available, then there is the need to resort to lower level mechanisms of the operating systems, such as RPCs, sockets, etc.
- Agentification of components, i.e. the transformation of each manufacturing component into an agent, to be part of a multi-agent system. The global systems integration can then be conceived in terms of a multi-agent environment (e.g. JADE (Bellifemine et al., 1999), FIPA[12] specifications and their experimental implementations, e.g. the FIPA Operating System (FIPA-OS) (Camarinha-Matos et al. , 1997; Suesmann et al. , 2002).

[12] Foundation for Intelligent Physical Agents (see http://www.fipa.org, where all specifications are available on-line)

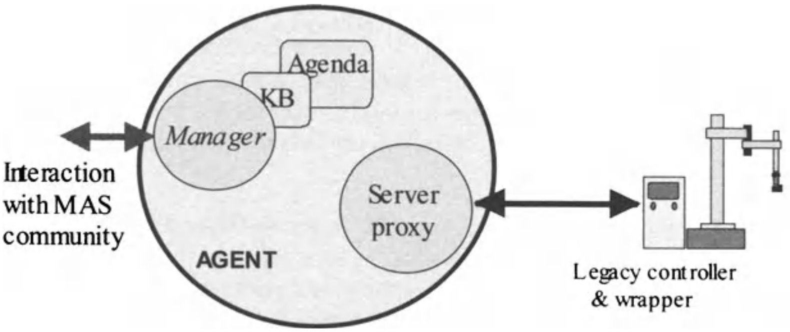


Fig. 18.11. Agents in manufacturing systems integration

- When the objective is to build an intelligent supervision system or a decision support system, reactive programming methods (i.e. daemons[13] attached to slots, in frame-based systems) represent an interesting mechanism for hiding the specificities of the invocation of services provided by manufacturing servers. In this way, it is easy to build a hybrid knowledge-based system that interacts with legacy components.

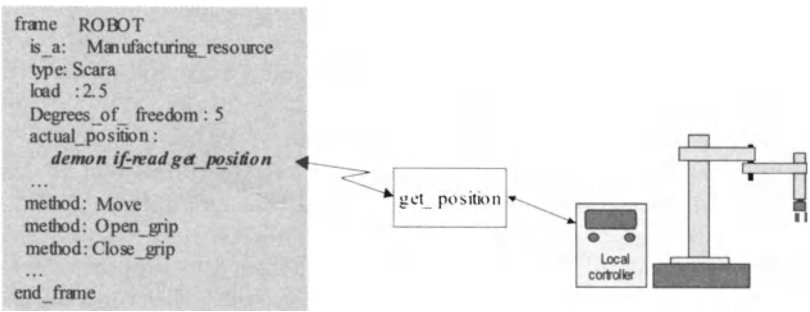


Fig. 18.12. Reactive programming in systems integration

Fig. 18.12 illustrates this process. If during its reasoning process a decision support system, for instance, tries to access the value of the attribute (slot) *current-position* of the frame (object) *ROBOT*, the daemon *if-read get-position*, which is declared as a daemon reactive to read operations, will automatically fire and the corresponding function will collect the value from the external source (robot controller).

[13] traditionally understood as limited functionality program modules running in the background and being triggered into action by time lapse or events.

18.3.3 MAS Approaches

This section focuses further on the agentification technique and in addition exemplifies it.

18.3.3.1 Manufacturing Resources as Agents

Here, two main approaches are suggested for agent encapsulation in agent-based manufacturing systems. Namely, the *functional* decomposition and *physical* decomposition (Shen , 2001). In the functional decomposition approach, agents are used to encapsulate functional modules (applications) such as order acquisition, planning, scheduling, material handling, transportation management, and product distribution, etc. In this case there is no explicit relationship between agents and their physical resources. In the physical decomposition approach however, agents are used to represent entities in the physical world, such as workers, machines, tools, fixtures, products, parts, operations, etc. Hence, there is an explicit relationship between an agent and its physical resource.

According to this second approach (perhaps the most popular) a first step in designing a MAS-based approach for shop-floor integration is the agentification of the manufacturing resources. This step involves two sub-activities: transformation of the resource controller into a server (wrapper building) and creation of an abstract representation (i.e. a *server proxy*) of the resource in the MAS environment. The idea is to provide a smooth integration between the functionalities provided by the proxy and the real actions performed by the resource controllers. The functionalities represented by the proxy are accessed using, for instance, Remote Procedure Calls (RPCs).

In the MAS environment each resource is then represented by an agent possessing a number of behaviors, including:

- Agent management – a behavior responsible for interacting with the MAS environment and acquiring new jobs for the resource. Acquired jobs (via contract net protocol, for instance) are recorded in the agent's agenda;
- Task execution – a behavior that is responsible for the application of the resource's skills, accessible via the corresponding proxy, according to the tasks assigned to the agent's agenda.

In terms of implementation, these behaviors may be materialized as two threads, as in the case of the JADE development platform, or as two specialized auxiliary agents working in tandem.

In order to discuss the agents themselves, it is important to make clear the distinction between **agent** and **component**. For instance, the model of a **robot component** is a description of its static and dynamic characteristics independent of the context, while a **robot agent** is a model of a robot and its associated resources (like the tools or auxiliary sensors) when inserted in a particular context. A robot agent can play different roles in different contexts.

For example, the expected behavior of a robot agent in an assembly context is different from its behavior in a spot welding context.
From the architecture point of view, different levels of agents can be considered on the shop floor.
A *first* level may be composed of those agents that directly interface to the machines. These agents have to carry out low level operations by loading the appropriate programs into the controllers or invoking remote services and observing their behavior and giving orders to them. These agents may interact in order to generate aggregated functionalities that in some cases are more complex than the simple addition of their individual capabilities. This is what happens, for instance, when several manufacturing components are working together on a manufacturing cell.
Therefore, the *second* level of agents can be composed in machine/device teams. A team or **consortium** is a set of strongly-coupled machines working together in order to achieve some complex result. The teams can be static (such as a cell), or dynamically formed, based on the partial skills of a set of agents from the first level.
A *third* level of agents may exist, which includes high-level agents responsible mainly for the supervision activity in the shop floor and tele-operation and/or tele-maintenance activities.

18.3.3.2 Flexible Coordination

In terms of coordination, simple architectures can be built with a fixed strategy, usually following a hierarchical structure, according to the specific roles of each agent.
For more complex systems, namely in shop-floors where alternative resources can be used for the realization of a given task, a highly flexible coordination can be achieved through *negotiation*. For this purpose, the Contract Net Protocol or its modified versions are commonly utilized. This protocol was first proposed by Smith (1980) and demonstrated on a distributed sensing system. The basic idea is that each agent (manager) having some work to subcontract, decomposes it into subtasks and, for each subtask, broadcasts an offer and waits for other agents (contractors) to send their bids. After some delay, the best offers are chosen and contracts are allocated to one or more contractors that process the subtask. This protocol leads to a coordination method for task allocation, providing dynamic allocation and "natural" load balancing. However, when the number of nodes is large, the number of messages on the network increases, which can lead to a situation where agents spend more time processing messages than doing the actual work, or worse, the system stops through being saturated by messages. A number of solutions have been suggested to overcome this difficulty. A typical solution considers agents to be organized according to their skills, and the task offers are sent only to the group possessing the appropriate set of skills.

An example in this direction is the HOLOS (Rabelo and Camarinha-Matos , 1994) system, developed for agile scheduling in manufacturing and that considers the following classes of agents:

- Scheduling Supervisor (SS): instance agent that performs the global scheduling supervision and is the unique system's "door" to other systems;
- Enterprise Activity Agents (EAA): instances that are directly associated to the manufacturing resources (robots, CNC machines, etc.), representing them into the agents community. These agents are the real executors of manufacturing orders;
- Local Distribution Centers (LDC): instances that represent functional clusters of EAAs in order to avoid global announcement broadcasting. Such groupings constitute a form of agent federation. They are also responsible for selecting the most suitable agent for a certain order, after a negotiation process;
- Consortium (C): temporary instance created to supervise the execution of a given order by the involved EAAs.

Figure 18.13 shows a general scenario of a particular HOLOS scheduling system.

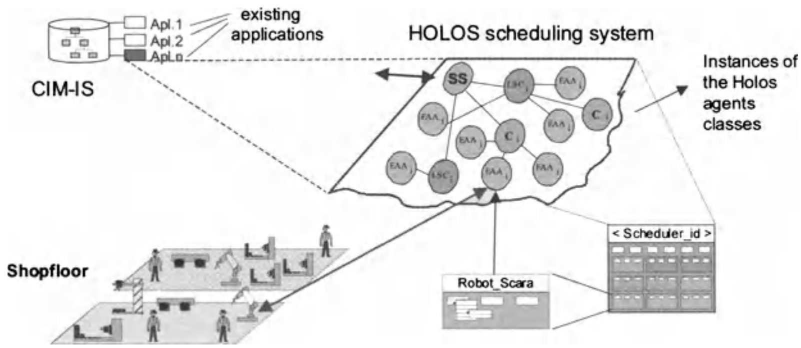


Fig. 18.13. - Example of a general MAS scenario

Once launched in the computational environment, all instances created by HOLOS become persistent, except the Consortium agents. One Consortium is alive and active as long as the task it supervises is not completed. Once the task is finished, the corresponding Consortium dismantles itself after generating some log information for auditing purposes. In HOLOS, there is no unique global and comprehensive schedule, but rather a collection of distributed and inter-related pieces of schedules.

HOLOS uses the Contract-Net Protocol coordination mechanism to support the task assignments to agents, and the negotiation method to overcome conflicts taking place during the scheduling process.

Figure 18.14 illustrates how the negotiation approach is used in HOLOS. Note that the main function of a scheduling system is to assign tasks (production orders) to manufacturing resources (robots, CNC machines, workers, etc.), represented by agents, during certain periods of time. The basic procedure consists of:

- announcing a task, i.e. an "enterprise activity", which is modeled as an object and represents the requirements of an individual process plan operation, through the MAS network;
- having the agents exchange information about it with other agents;
- selecting an agent to perform such task at the end of this process.

(a) Task announcement (b) Bidding and selection

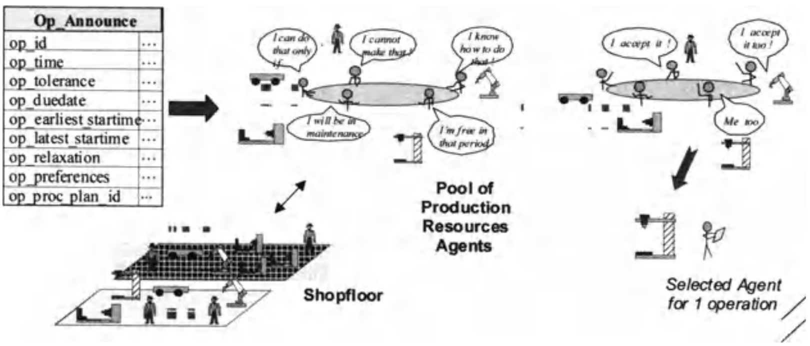


Fig. 18.14. Negotiation on scheduling.

Although this basic Contract Net and its variants are usually used as negotiation protocols in most agent-based scheduling systems, market-based approaches (in which multiple-step negotiation and bargaining mechanisms are applied) are becoming more and more popular.
In terms of development environments, during the last decade a large number of platforms (many of them open source) have been proposed. A very important de facto standardization initiative is represented by FIPA, which aims at normalizing the architecture of multi-agent systems and the inter-agent communication language (ACL). Various open source platforms (e.g. JADE, FIPA OS) are FIPA compliant.

18.3.4 Towards Agile Organization and Emergent Behaviour

The capability to rapidly change the shop-floor infrastructure is an important condition to allow participation of manufacturing enterprises in dynamic cooperative networks. Shop-floor agility implies an increasing level of re-engineering activity. The processes of change (re-engineering/adaptation)

have been addressed mostly at the level of business process re-engineering and information technology infrastructures but little attention has been devoted to the changes of the manufacturing system itself; yet, the shop floor undergoes a continuous evolution along its life cycle.

A particularly critical element in the shop-floor re-engineering process is the control system. Current control/supervision architectures are not agile because any shop-floor changes require considerable programming modifications, which imply the need for qualified programmers, usually not available in manufacturing small and medium enterprises (SMEs) . To worsen the situation, even small changes may impact the global system architecture, which increases the programming effort and the potential for side-effect errors. It is therefore mandatory to develop approaches to eliminate or reduce these problems, making the process of change (re-engineering) faster and more flexible e.g. by focusing on configuration instead of software coding.

Contract-based reengineering. An agent-based architecture in which co-operation is regulated by contracts has been proposed by the Robotics and Integrated Manufacturing Group of UNL (Barata and Matos , 2002b) as a flexible approach to dynamic shop-floor re-engineering. In this approach, each manufacturing component (e.g. robots, tools and fixing devices) is associated to an agent that represents its behavior. A set of agents working together towards a common goal is called a ***consortium***. The consortium (the agency equivalent to a manufacturing cell) is the basic organizational form of co-operation. A basic consortium is composed of one or more agents performing the role of *manufacturing agents* and one agent performing the role of *coordinator*, and can also include other consortia as members. The coordinator of a consortium is able to execute complex operations (it possesses complex skills) that are composed of simpler operations offered by the consortium members. It is important that agents willing to cooperate are grouped according to their spatial relationships (or any other relevant relationship - e.g. technological compatibility), meaning that manufacturing agents that could establish consortia should be grouped together (because they share something when they are candidates to consortia). The structure used to group manufacturing agents that can co-operate and from which the agents share some concepts is called manufacturing *cluster*. Adding or removing a component from the physical manufacturing system also implies that the corresponding agent must be removed from the cluster, what can also have impact on the established consortia. Each consortium is formed with the help of a *broker*. A broker is an agent that interacts with a human, the cluster, and candidate members to the consortium.

The interactions between the cluster and its members are regulated by a ***cluster adhesion contract***, which is a bilateral contract formed between the cluster and the candidate member. This contract establishes the terms under which the co-operation is going to be established. It includes terms such as identifying the ontologies that must be used by the candidate, the duration

of the contract, the consideration[14] (Barata and Matos , 2002a). The behavior of the consortium is regulated by a ***multilateral consortium contract*** that is "signed" by all members of the consortium. The important terms of this type of contract other than the usual ones (such as duration, names of the members, penalties, etc.) are the consideration and the individual skills that each member brings to the consortium. Note that the skills involved in a specific consortium contract may be a subset of the skills offered by the involved agent when it joins the cluster. The importance of contracts as a mechanism to create/change flexible and agile control structures (consortia) lays on the fact that the generic behaviors presented by generic agents are constrained by the contract that each agent has signed. This calls forth that different consortium behaviors can be achieved simply by changing the terms of the consortium contract, namely the skills brought to the consortium.

An agent can participate in a consortium according to two different roles: as member or as coordinator. A member must execute all the operations signed for in the consortium contract that are requested by the coordinator. On the other hand, the coordinator can create complex operations (services) by aggregation of the individual operations of the consortium members. A coordinator can have more complex operations either by adding more members or through the creation of a hierarchical structure of consortia.

It is interesting to note that in both roles agents might be subject to two kinds of contracts: ***membership contracts***, and ***coordination contracts***. This expresses the dual role of the agents (since an agent can be simultaneously a member of one consortium and the coordinator of another consortium). This is the case of the agent Z for instance, in Fig. 18.15 that coordinates members X and Y, and simultaneously is a member of another consortium led by R. The terms written on the right side of each contract in Fig. 18.15 represent the skills offered by the agent to the consortium. For instance, agent Z knows from its coordination contract that it can access the skills SA and SC, provided by agent Y, and skill SB, provided by agent X. From these skills agent Z creates the more complex skills SD and SE, which are offered to a higher-level consortium contract, which are represented in the membership contract of Z. The services (complex operations) of each level can be represented using a workflow model or even a Petri net, and each generic agent must have an execution engine to execute these kinds of operations. In the current implementation these operations are created with the help of a human expert interacting with the broker agent, during the creation / evolution of a consortium.

Fig. 18.15 also shows that a ***manufacturing agent*** coordinates an AMI (Agent Machine Interface), which is an agent representing the controller of some specific physical manufacturing component.

---

[14] a term which describes what the candidate should provide in turn to join the cluster (usually the skills or capacities that the candidate is bringing to the cluster)

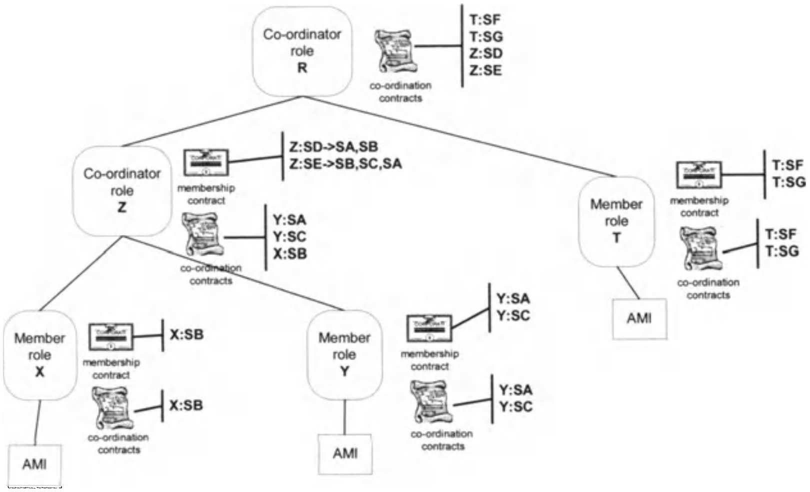


Fig. 18.15. Hierarchy of consortia

The most important global behaviors of the proposed architecture can be summarized as: 1) cluster registering, 2) new consortium formation, 3) consortium alteration, 4) service execution, and 5) contract termination.

18.4 Intra-Enterprise Infrastructure Design

18.4.1 Brief Historic Overview

Enterprise integration is usually understood as the process of connecting and combining people, systems, and technologies to guarantee that the right information and resources are available in the right place at the right time in order to facilitate the accomplishment of the enterprise's business objectives. The IT infrastructure is naturally a key element in this process.
The rapid pace of changes in the underlying enabling technologies has generated a lot of confusion and a serious obstacle in the companies' investment in integration technologies. In order to reduce risks, enterprises need to use flexible and standard-based architectures with inter-operable components.
The intra-enterprise integrating infrastructures are closely related to the enterprise reference architectures, since the reference architecture defines how the various subsystems of an enterprise, e.g. engineering, manufacturing, marketing, product support, etc. interact from the functional and operational perspective.
During the 1980s and 1990s several reference architectures were developed that contributed to a better understanding of the enterprise integration problem. For instance, CIM-OSA (AMICE , 1993) was one of the first models

to suggest an Integrating Infrastructure (Fig. 18.16) that ideally should support a kind of enterprise-wide "plug-and-play" for resources and applications, while providing a kind of enterprise operating system able to seamlessly run the business processes. Unfortunately, the supporting modeling tools and plug & play infrastructure for the CIM-OSA model have not fully materialized as expected.

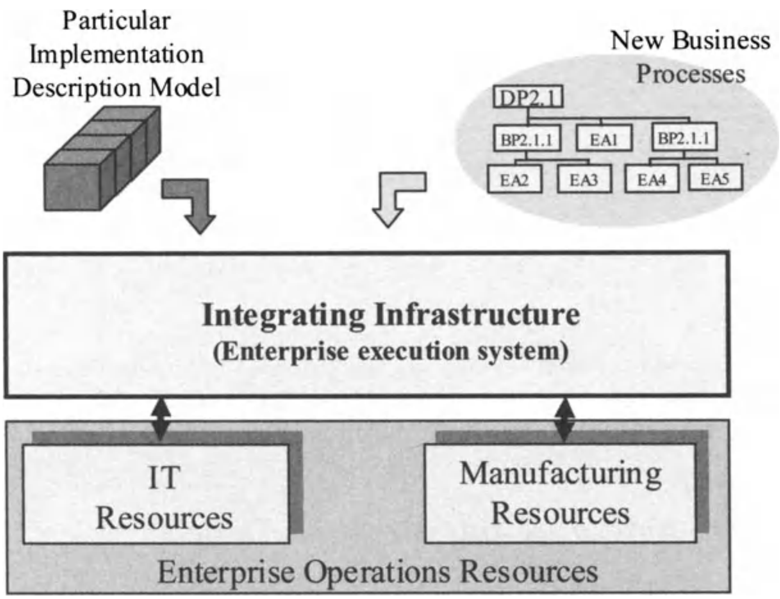


Fig. 18.16. The CIM-OSA Integrating Infrastructure model

During the last two decades several other projects focused on introducing reference architectures, and have identified the need for an enterprise integrating infrastructure. That is the case, for instance, of Generalised Enterprise Reference Architecture and Methodology (GERAM, (IFIP-IFAC , 1999)) and the CASA[15]/SME[16] Manufacturing Enterprise Wheel model (CASA , 1993).
To be useful, in addition to complete definition, reference architectures need to be accompanied by appropriate engineering tools for their planning, modeling, simulation, and analysis. However, the lack of such support tools causes the generic reference architectures to rarely be used as more than a starting point for enterprise integration. Most systems in practice have been developed more on an ad-hoc basis.

[15] Computer Automated Systems Association
[16] Society of Manufacturing Engineers

That said, it must be noted that some of the reference architectures purposely remain neutral by not prescribing any specific tool but rather specifications for such tools (e.g. PERA, GERAM). Other reference architectures are supported by third-party tools[17] or by software produced by commercial spin-offs of the (mostly academic) groups that produced the architectures[18]. The reader is referred to Chapter 3 for more concrete examples of tools supporting specific architectures.
In parallel with a better understanding of the functions of the enterprise, the matching software applications were refined / generalized and extended to progressively support more and more business functions. From simpler software applications such as: manufacturing resource planning (MRP), or computer aided design (CAD) and process planning (CAPP) systems, there was a rapid evolution to large monolithic systems such as production planning and control (PPC), enterprise resource planning (ERP), and product data management (PDM) systems. More recently, other areas have gained more importance and new tools or further extensions to existing tools are continuously being added (e.g. SCM [19]).
The monolithic solutions, representing large "islands" of integration, in addition to the high costs, introduce a number of difficulties in terms of configuration / adaptation costs, integration of other components, etc. Nevertheless, as they represented an integrated solution for a large part of the enterprise needs, they had a wide expansion and acceptance during the 1990s.
In order to avoid the interoperability difficulties, and in parallel with the modularization "fashion" of the software components paradigm, some timid efforts were invested in the direction to decompose or "fragment" the monolithic enterprise applications. This was the case of the Open Applications Group initiative (OAG , 1997) that aimed to create common standards for the integration of enterprise business applications (Fig. 18.17).
The OAG specifications seem to be complementary to other interoperability work currently underway by other standardization initiatives such as the Object Management Group (OMG), but their practical impacts are yet to be determined.
Presently, the intra-enterprise integration is assisted by several tools and technologies. The current state of the support infrastructures for enterprise integration is characterized and summarized in (Integrated Manufacturing , 1999):

---

[17] such as e.g. CIMOSA with the FirstStep tool from Interfacing Technologies, or C4ISR with the System Architect with C4ISR option, produced by Popkin Software (www.popkin.com) or FrameWork, produced by Ptech, Inc.(www.ptech.com)

[18] e.g. IMAGIM produced by GRAISoft (www.graisoft.com), in the case of GRAI-GIM.

[19] Supply Chain Management

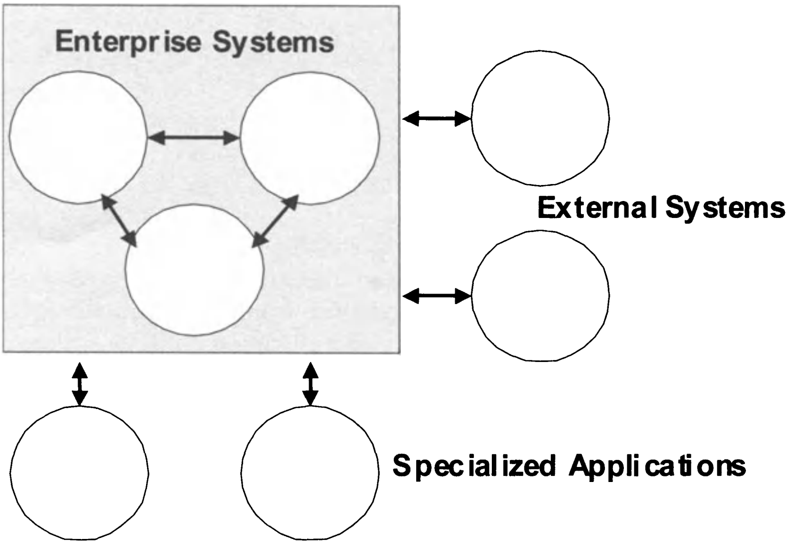


Fig. 18.17. The OAG componentization and integration framework

- Enterprise information systems have moved from large mainframe computers to distributed client-server, PC-based systems.
- Although still not fully user-friendly, the underlying integrating infrastructure is progressively becoming more transparent to its users.
- In most companies, all computers and much of the manufacturing equipment are now connected through LANS and WANS.
- There is a wide range of standards affecting data exchange, interfacing formats, and terminology.
- Interoperability has been achieved in some functional areas (e.g. EDI, product data interchange with IGES or STEP) in technology intensive sectors (e.g. automotive, aerospace), but the situation is different in other sectors.
- In some industry sectors, much of the base software is customer-built. Additionally, lots of valuable data are still confined to legacy applications.
- The use of Internet as a collaboration tool and information exchange mechanism is growing dramatically.
- Web-based networking is adding "thin" web-based clients to provide more open access to the enterprise functions.
- Plug-and-play modules (plug-ins) are now common for some PC applications, but not for complex business systems where cross-platform, cross-application compatibility is still a major problem.

18.4.2 Information-based Integration Design

18.4.2.1 Loose and Tight Integration

Enterprise-wide integration of information has been one of the first focus points in enterprise integration.
Simple approaches, corresponding to a form of *loose integration*, are based on file interchanges among applications. This usually requires pair-wise translations between applications in order to make the semantics and syntax of data generated by one application understandable by the other. This is a common practice when integrating some legacy applications with a monolithic application such as an ERP system. But, when the aim is to integrate a large number of subsystems, using a pair-wise translation is quite inefficient implying the development of 2 translators for each pair of interacting subsystems.
An alternative to this method is to adopt a *common intermediate information interchange format*. In this case, for every application there is the need to only create one pre-processor (translating from the application to the intermediate format) and one post-processor (translating form the intermediate format to the internal format adopted by the application). This approach is materialized for specific sectors and has been widely explored, for instance, in the area of CAD / product data interchange (e.g. IGES, DXF, or STEP neutral formats). But still its adoption enterprise-wide would be a huge task.
A third information integration approach, *tight integration*, corresponds to the adoption of an information management system that handles the data to be shared, and the various subsystems interact, through some query language, with this system. Centralized, distributed and federated approaches to implement tight integration have been developed.

18.4.2.2 Centralized Information System

Early approaches to enterprise information integration were mostly centralized. The dream was to have a central data repository, the so-called Enterprise Information System, where all relevant data generated and used by the various units of the enterprise reside i.e. physical integration. The concept of CIM was strongly influenced by this approach and a typical enterprise representation considered the information system as the core system around which the various departments were positioned, as illustrated in Fig. 18.18.
According to this model, various views of the data, both in terms of the information structures and the information visualization forms, would be derived from the structures of the central repository. The data warehouse concept represents a variant of this old idea.
One of the design issues here has to do with the fact that most common technologies for information management (e.g. relational DBMSs) were designed due to the needs of the traditional management applications. Engineering and Manufacturing systems integration imposed new requirements such as:

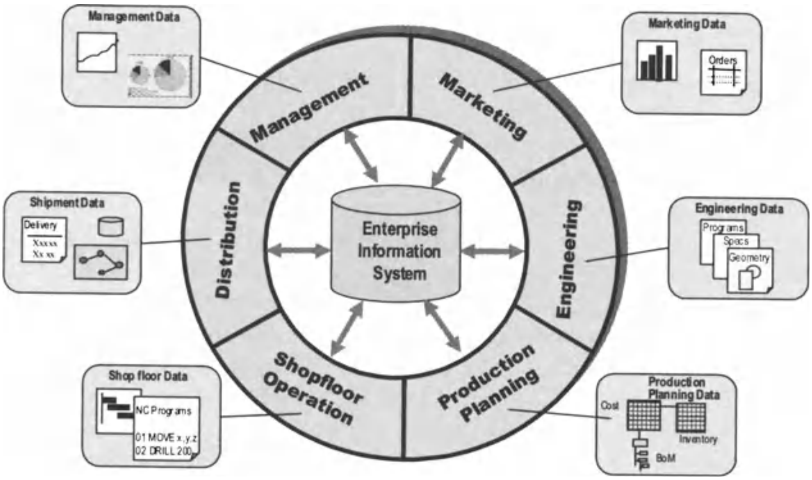


Fig. 18.18. A high level enterprise reference model based on a centralized information system

- Large number of entities with small number (in most cases) of instances per type
- Complex inter-relationships between entities
- Dynamic grow of the data description (schema)
- Some transactions might need random accesses over large volumes of information and might have long temporal spans, introducing questions of exclusivity and concurrency.
- Information visibility support and view definition for individual users are required to any response to authorized queries.
- Support for basic types (e.g. integer, real, etc.) and complex types (e.g. points, lines, matrixes, etc.) are required.
- Support for sub-spaces and their complex relationships (procedural, functional, temporal, positional) are required.
- Versioning needs to be supported and managed.
- Cooperative / joint use of entities - Several engineers may work concurrently on the same project (Concurrent Engineering), etc.

Some of these requirements are already addressed in different technologies (e.g. specialized CAD DB, KBMS) but no single technology, alone, covers all of these various needs of the enterprises. Two possible directions can then be pursued (Fig. 18.19): either (i) each technology borrows some concepts and mechanisms that have been introduced and proven by other technologies or (ii)

an interoperation among the various technologies is attempted for the base infrastructure in order to reach a logical multi-system information management system (IMS). Software developers, in an attempt to dominate the market, naturally tend to follow the first approach. The reality of enterprises where legacy systems include a diversity of technologies would certainly suggest and benefit from the second approach. The federated information management paradigm mentioned below offers one solution for the latter case.

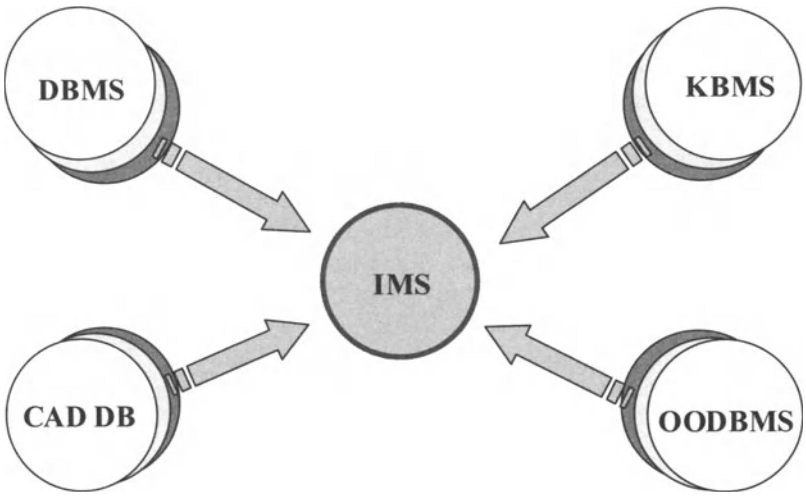


Fig. 18.19. Combination of different technologies for information management

18.4.2.3 Distributed / Federated Information System

By mid 1990s it became evident that, in spite of all efforts and standardization initiatives, the success of centralized information systems would be limited in most cases. This is justified by a number of factors, including:

- Technical factors: diversity, volume, complexity and volatility of the information to be managed enterprise-wide make it rather difficult to design and maintain a central information system.
- Technological reality factors: the spread of PCs and local has networks created the conditions for every sector to be able to manage its own information.
- Social factors: the feeling of power associated to the ownership of information does not make information sharing a natural tendency.

Various approaches addressing information exchange among heterogeneous legacy databases (e.g. ODBC) have facilitated the practical use of distributed

information management and multi-database. A distributed database can be defined as consisting of a collection of physically distributed data in separate databases running on independent computer systems, whereas the control of the distributed database system and the schema definition is common and centralized. These computers are interconnected through the network and each system has autonomous processing capability serving local applications. A variety of multi-data systems further support the integration of data from different DBMSs. Each system may in fact participate in the execution of one or more global applications. Such applications require data from more than one site. The distributed nature of the physical data shall be hidden from users and this transparency manifests itself in a number of ways, i.e. location transparency, replication transparency, transaction transparency and catalog transparency.

Furthermore, one of the most promising multi-database solutions is the design of a *federated* information management system that facilitates reaching a good balance between the desire for autonomy (that is natural in each unit of the enterprise) and the advantages of sharing some information with certain authorized users, in order to improve the overall performance of the enterprise. According to this approach, the enterprise can be seen as a network of distributed, autonomous, and heterogeneous nodes, where nodes represent various units of the enterprise (Fig. 18.20).

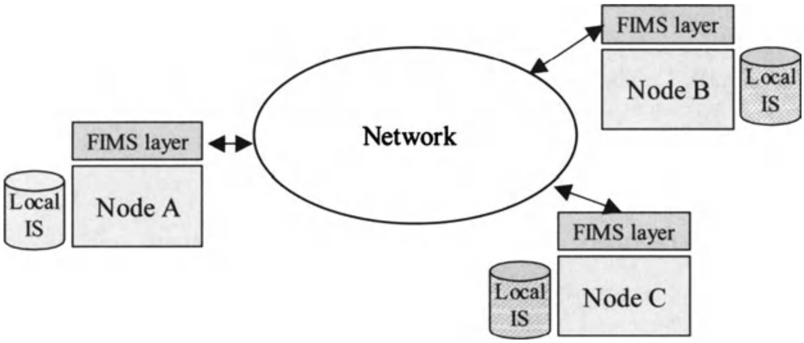


Fig. 18.20. The enterprise as a federation of nodes

Each federated node has its own computational power, runs its own applications, and is responsible for its own local data. In order to allow sharing and cooperation, each node is complemented by a layer – the Federated Information Management System (FIMS) layer – that allows for controlled sharing of information among nodes. Figure 18.21 shows an example architecture of such federation layer, from the PEER system (Afsarmanesh et al., 1994).

In PEER, interdependencies between two nodes' information are established through their independent conceptual schemas defined on their information.

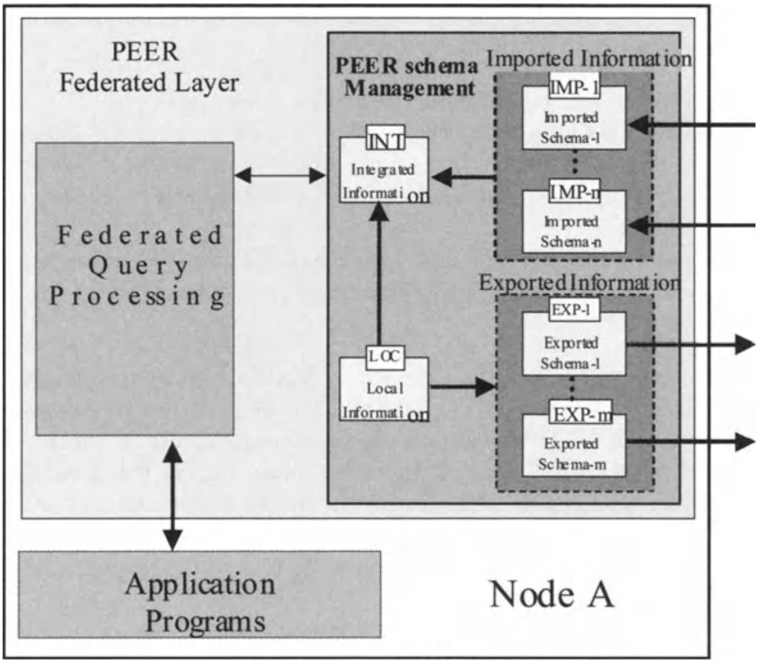


Fig. 18.21. Example of Federated Information Management System

Every node is represented by several schemas:

- a local schema (LOC): the schema that models the data stored locally;
- several import schemas (IMP): the various import schemas model the information that is accessible from other databases;
- several export schemas (EXP): an export schema models some information that a database wishes to make accessible to other nodes (usually, a node defines several export schemas);
- an integrated schema (INT): the integrated schema presents a coherent view on all accessible local and remote information. The integrated schema can define a particular classification of objects that are classified differently by the schemas in other nodes.

Every node is able to import data from other nodes through their import schemas, via the federated query processor, and access their data according to the bilaterally pre-defined access permissions. This approach corresponds to a kind of *virtual integration* in the sense that it gives the user the *impression* of working with a single information management system while, in fact, the data is actually managed by several independent and autonomous systems, and there is no centralization / replication of data or control. Some of the

functionalities supported within the PEER federated database environment are highlighted below:

- objects stored in a node can be shared with other nodes;
- when needed up-to-date objects can be accessed remotely from other nodes;
- different access privileges and information visibility rights can be defined for other nodes;
- transparency of the physical and logical information distribution among the nodes supported through the federated query processing access to information.

As a consequence of this general interaction facility, the federated approach allows for cooperation between the federated nodes, in order to accomplish a common global task, while the autonomy and independence of every node is preserved and reinforced. Of course there is a need, during the design phase, to negotiate pair-wise between the nodes the actual information that will be shared between them. This bilateral process is however likely to be easier to accept by the "owners" of the information than a solution that requires a centralized repository.
Presently, one of the difficulties of this approach is the lack of off-the-shelf products developed according to this paradigm. Several implementations have been made at research level, but the practical application still requires the systems engineer to develop a system's own federation layer.

18.4.2.4 The Role of Standardization

An open and flexible solution towards a generic approach for information exchange and integration must apply standards as much as possible and suggest new standards when they are not available. Currently, several standard tools and mechanisms can be used for this purpose. For instance, consider the following:

- From the database development perspective, one option is for instance to use ODL (Object Definition Language) to support the portability of database schemas across conforming ODBMSs. OIF (Object Interchange Format) and XML can also be used to exchange objects between databases and provide database documentation. Simultaneously, independent access facilities among the data sources can be supported through the standards and middleware solutions (e.g. ODBC, JDBC, and JAVA).
- From the users, applications, and database accesses perspectives, a combined solution can be based on standard interfaces and languages (e.g. SQL, OQL, Java, and C++), in order to:
  - Provide the requested information for users and applications,
  - Facilitate the access to heterogeneous and autonomous underlying data sources,

- Support flexible access to those underlying data sources based on standard technologies and via secure and reliable export schemas mechanisms,
- Support multimedia information and large data sets.
- ...

The design of common data models and ontologies is one of the most important challenges in systems integration. Even when the channels and basic communication protocols are in place, the information sharing among subsystems is frequently affected by the fact that the semantics of information depends on the context in which it is considered. This is particularly true in manufacturing enterprises, due to the growing complexity of the information to be handled and the growing need to exchange information among the various application programs.
One way to overcome this difficulty would again be the adoption of common norms and standards. For some areas this goal can be achieved. This is the case of exchange of technical product data (e.g. through the STEP standard) or the exchange of business information (e.g. through EDIFACT). Some other initiatives have tried to combine these two standards (e.g. the inclusion of STEP models in the CONDRA' drawing administration' messages of EDIFACT). But even for these two specific areas, several aspects are not covered by the standards. Thus, there is a need to cope with different versions of the standard and to configure it to each application context.
Most other industry areas however, still lack common standards for definition of their concepts and entities.
It shall also be noted that the process of elaboration and approval of standards in the international standardization organizations (e.g. ISO) takes a very long time. This is mostly inadequate to the speed of the evolution in IT. As a reaction to this situation, some "private" initiatives for standardization (led by large ad-hoc organizations of software developers) have emerged. One of the current "fashionable" approaches for information exchange is the eXtended Markup Language (**XML**). Very rapidly, since its ratification by the World Wide Web Consortium (W3C) in 1998, XML is becoming a *de facto* standard for *data communication and information exchange* among distributed organizations and/or applications in the same organization. Thus, one of the main applications that benefits from XML are information exchange and data integration among heterogeneous databases. Similar to the ODMG[20] and Java standards, the advantages of using XML standard is that it helps to reduce the number of developed wrappers to serve the interoperation among heterogeneous and distributed databases and applications. Integrating XML data across applications and databases is of great interest for the information systems community. Thus, efficient techniques for integrating XML data across local- and wide-area networks are an important research focus.

[20] Object Data Management Group

However, the consideration of XML for database and in particular for *data integration*, raises several challenges for database research. In fact, using XML as the standard *syntax* for data exchange and representation only partially advances the prospects of their integration. To support data integration at the *semantic* level however, there must first be a certain type of agreement, at least on the format and meaning, among all involved database / application nodes with specific Document Type Declarations (DTDs) in XML. For instance, if the XML data does not conform to a fixed schema, names and meanings of the used tags are arbitrary and the data is self-describing, i.e. no semantics can be derived from the structure in XML documents. There are therefore several issues that need to be considered when designing an XML-based integration approach:

- Languages for describing the contents of XML sources, which provide the semantic mapping between the data in the source and the relations in the mediated schema. The main challenges involve (1) the restructured data appearing in XML is richer than in relational data, (2) scaling up to a very large number of XML sources must be supported, and (3) exploiting the knowledge conveyed by accompanying DTDs is required.
- Query reformulation algorithms, which require the development of algorithms for efficiently rewriting user queries (posed on a mediated schema) to queries that refer to different underlying XML data sources. The known techniques for reformulation from the relational case do not extend easily to languages for querying XML documents.
- Translation among DTDs, which provides the proper tools and facilities to translate XML data conforming to one DTD into an XML document conforming to a different DTD, presumably with semantically related content.
- Obtaining source descriptions, which develops methods for automatically (or semi-automatically) computing source descriptions for newly introduced XML data sources. These methods become significant especially when the number of data sources grows.

Simultaneously, developments in the knowledge management area, trying to support the creation of **common ontologies**, represent a promising approach to handle the semantic aspects. A number of tools, some freeware (e.g. Protégé), facilitate the process of building and managing ontologies and are likely to play an important role in future systems integration.
Nevertheless, the difficulties of reaching consensus among the various parties in order to define a commonly accepted ontology for a particular sector should not be underestimated.

18.4.3 Knowledge-based Integration Design

Intelligent systems can play an important mission in the management and mission support of the enterprises. Decision support, automatic or interactive

planning, supervision and monitoring, are just some examples of systems that are progressively entering the enterprises world.

After the boom of the expert systems / knowledge based systems (KBS) of the 1980s, there was a retraction in terms of the amount of efforts put in this area. One of the reasons of the relative failure of early expert systems when applied to real enterprise systems was their isolation from the legacy systems. The KBSes' isolation from the enterprise information sources, (irrespective of how sound the technology was) has resulted in the development of early KBSes as simple toys with limited value.

The obvious solution is to build *hybrid* systems, combining a knowledge-based component (knowledge base + inference engine) with other "traditional" systems, as shown in Fig. 18.22. Furthermore, although KBSes are powerful in terms of modeling expressiveness (frames, rules) and reasoning, they are weak in terms of data persistence (working mainly in the computer's RAM[21]) and handling large amounts of information (topics that have been answered by classic DBMSs). They are also weak in terms of the integration of data from distributed heterogeneous and autonomous sources (topic covered by the distributed / federated databases).

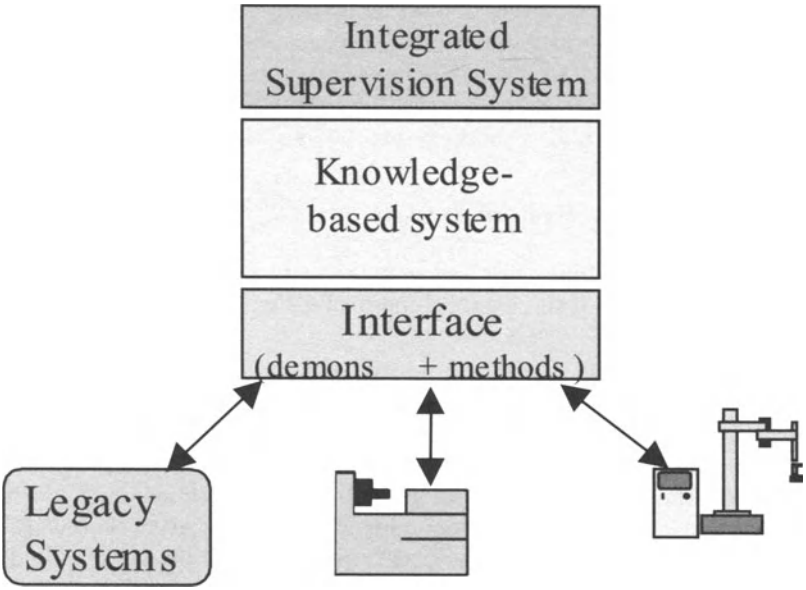


Fig. 18.22. Hybrid knowledge-based systems

[21] Random Access Memory, consisting of elementary binary memory cells which typically need a permanent power supply and also to be periodically refreshed in order to keep their data content intact. The advantage of RAM is mainly the speed compared to other, e.g. persistent type of data storage.

The integration can be attempted in two ways both including the development of a KBMS (Knowledge Base Management System):

- "Static" (or loose) integration: the data is imported from DBMS. This requires the manual building of an ontology on the KBMS side and implementing a mapping mechanism for importing the data (querying) from the DBMS. The imported data will generate instances of the ontology concepts. This import process is conducted initially, before any reasoning is started on the knowledge-based system side.
- "Dynamic" (or tight) integration – the KBMS interacts with the external data sources acquiring data when needed. External data remains under the control of the external sources (subsystems) and only when needed (e.g. during a reasoning process) specific data elements are imported. In this way, the knowledge-based system can always count on updated information.

Similarly to what was suggested in section 18.3.2, reactive programming, i.e. the association of daemons to frame slots that react when some operation (e.g. read, write) is attempted, provide a practical mechanism to implement a dynamic integration. A transparent mechanism for hybrid systems can be achieved by "hiding" daemons behind slots. This process however, requires legacy systems to be implemented under a client-server philosophy.
An alternative approach would be the use of methods to implement an explicit connection with the external systems. However, this approach would be less transparent, since methods would need to be explicitly called.

18.4.4 Coordination System Design

Once basic interoperability mechanisms are in place, the next challenge for enterprise integration is the establishment of a flexible activity coordination system.
Coordination represents a new and multidisciplinary research area that finds its roots in contributions from many other disciplines such as computer science, organization theory, economics, linguistics, psychology, artificial intelligence, etc. In fact, in recent years, there has been a growing interest for coordination theory and coordination mechanisms (Malone and Crowston , 1994; Papadopoulos et al. , 1998; Schmidt and Simone , 1996; Simone et al. , 1996), namely when discussing the activities performed by complex and distributed systems. Different authors have proposed a number of definitions for coordination. Some of these definitions are:

- "Coordination is managing dependencies between activities" (according to Malone and Crowston (1994));
- "Coordination is the process of building programs by gluing together active pieces" (Carriero and Gelertner , 1992);

- "Coordination can be defined as the study of the dynamic topologies of interactions among interaction machines and the construction of protocols to realize such topologies that ensure well-behavedness" (Arbab , 1998);
- "Coordination is a property of interaction among some set of agents performing collective activities" (Bond and Gasser, 1988).

The development of coordination mechanisms in computer science started long before this area has shown signs of emerging as a quasi-autonomous discipline (Camarinha-Matos and Lima , 1999b). The early coordination mechanisms can be traced back to the initial works on operating systems and real time concurrent programming. Another important contribution can be found in the area of Petri nets and its many dialects, especially when this formalism started to be applied not only as a modeling and analysis tool, but also as a *control* mechanism, (by firing external actions as a side effect of a Petri net dynamic evolution). More recent contributions to coordination mechanisms can be found in the areas of Workflow Management Systems (WMS), Computer Supported Cooperative Work (CSCW), Business Process Modeling, Multi-Agent Systems (MAS), and Coordination Languages.

Petri nets represent a tool for modeling systems with interacting and concurrent components. It represents an analytic and graphical approach useful for modeling, analyzing and simulating discrete event dynamic systems. A Petri net Pn{P, T, I, O, M} is a directed graph consisting of two types of nodes, places (P) and transitions (T), where arcs are either from a place to a transition (I) or from a transition to a place (O). In graphical representation, places are drawn as circles and transitions as bars. A marking (state) assigns to a place a non-negative integer (number of tokens).

A place may represent a process or activity (or a state) and a transition may represent a change of state or the ending of activities that are input to the transition and the starting of activities that are output of a transition. One token in a place may indicate that the corresponding activity is being processed or that a resource is available in that state. The dynamic behavior of the system modeled by the Petri net can be captured by the transition firing rules. A transition is enabled to fire if there is at least one token in each place that is input to that transition. When an enabled transition fires, one token is consumed from each input place and a new token is sent to each output place.

With this basic model it is possible to represent precedence and synchronization dependencies between activities. It is also possible to have a hierarchical representation by expanding places (or transitions) into more detailed networks. A set of algorithms are available to analyze properties of the system modeled by the Petri net such as deadlocks, loops, reachability of a given state, etc.

Departing from the original formalism, a large number of extensions (intended to simplify the models or to support the representation of additional

attributes) have been proposed, such as Temporized Petri nets, Colored Petri nets (CPN), Predicate Transition nets, etc.
The use of Petri nets as a basis for defining control algorithms / coordination plans has led to the appearance of several high-level Petri net proposals, extending the basic formalism (Carriero and Gelertner , 1992). Invocation of a service application can be modeled as a side-effect of a transition (Camarinha-Matos and Lima , 1990).
Although there is a wide acceptance of Petri nets for modeling manufacturing systems, its use in enterprise integration is not a common practice. Some authors advocate its "superiority" compared to other formalisms due to its formal background, but its practical use in this domain is still quite limited, probably due to the existence of many variants of Petri nets and the lack of "standardized" software packages.
Although Petri nets are not directly used, the basic concepts and mechanisms proposed by this formalism can be found in the background of many other tools used in workflow and business process management.
Workflow management systems represent a popular approach to manage and support business processes. A workflow supports the automation of procedures, where documents, information or tasks are passed between participants based on a predefined set of rules in order to accomplish an objective (Lawrence , 1997). Although not based on a sound formal background as Petri nets, workflow systems have two major advantages that justify their popularity:

- the architecture of a workflow system explicitly emphasizes the interoperation between heterogeneous legacy systems, taking care of the communication among those systems.
- the supporting technology and tools have reached a quasi-standard status due to the joint efforts of the major software vendors through the Workflow Management Coalition (WfMC, in (WfMC, 1994)).

As a consequence, there is a well-established market for this technology, which justifies its wide adoption.
The WfMC's mission is to promote and develop the use of workflow through the establishment of standards for software terminology, interoperability and connectivity between workflow products. The WfMC reference model is composed of five key modules (Fig. 18.22). For each module, there is a set of functions that should be accomplished by the commercial products covering that module. The WfMC also defines the interfaces between the five modules. All workflow management systems have a number of generic components that interact using a set of predefined ways. Different products will offer different levels of capabilities within these generic components. To reach interoperability among various products it is necessary to adopt a standard set of interfaces and data interchange formats in those components, which is one of the contributions from the WfMC Reference Architecture (see interfaces in Fig. 18.22).

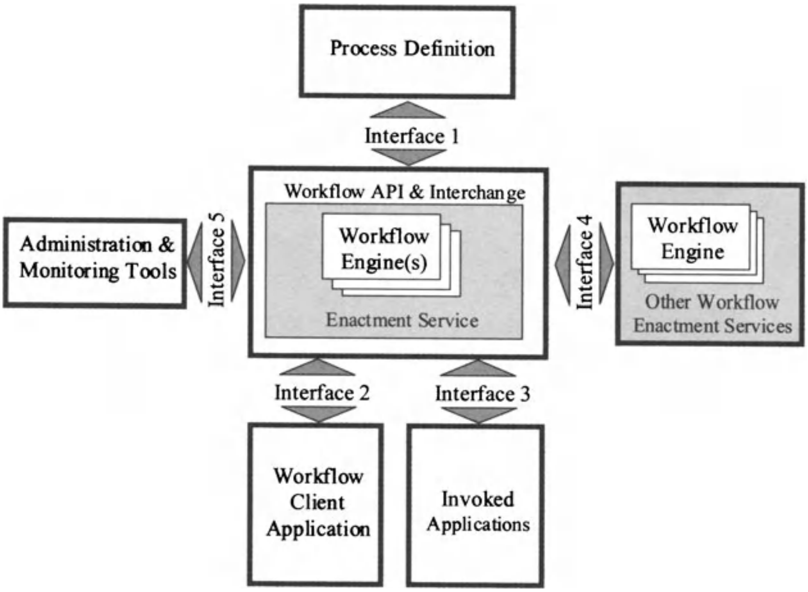


Fig. 18.23. WfMC Reference Architecture

Fig. 18.24 shows an example of a graphical editor for specifying the coordination processes. This graphical specification is then converted into the textual Workflow Process Definition Language (WPDL) (interface 1).

The workflow research community is also overcoming a major drawback, namely to cope with *dynamic* changes in workflow models. Good references can be found in the literature, such as Reichert et al. (1998), and the German project *"Knowledge-Based Dynamic Modification of Workflows"*, being developed at the University of Leipzig[22] (which presents good solutions to this problem).

Nowadays, the Business Process (BP) modeling and analysis area represents a very active and growing market for which a large number of tools are available. Some examples are: ARIS form IDS Scheer[23], Workflow Modeler and Analyzer by Metasoftware[24], IsModeler by Modus Operandi[25], Grade Modeler by the Grade Development Group[26], FirstSTEP by Interfacing Technologies[27], SIMPROCESS by CACI[28], etc. An attempt to standardize the business process

[22] http://www.informatik.uni-leipzig.de/ifi/abteilungen/db/Research/wf-modification.html
[23] http://www.ids-scheer.de
[24] http://www.metasoftware.com
[25] http://www.intecsys.com
[26] http://www.corpdyn.co.uk/Global.html
[27] http://www.interfacing.com, refer Chapter 3 for some details.
[28] http://www.caciasl.com/simprocess-product.html

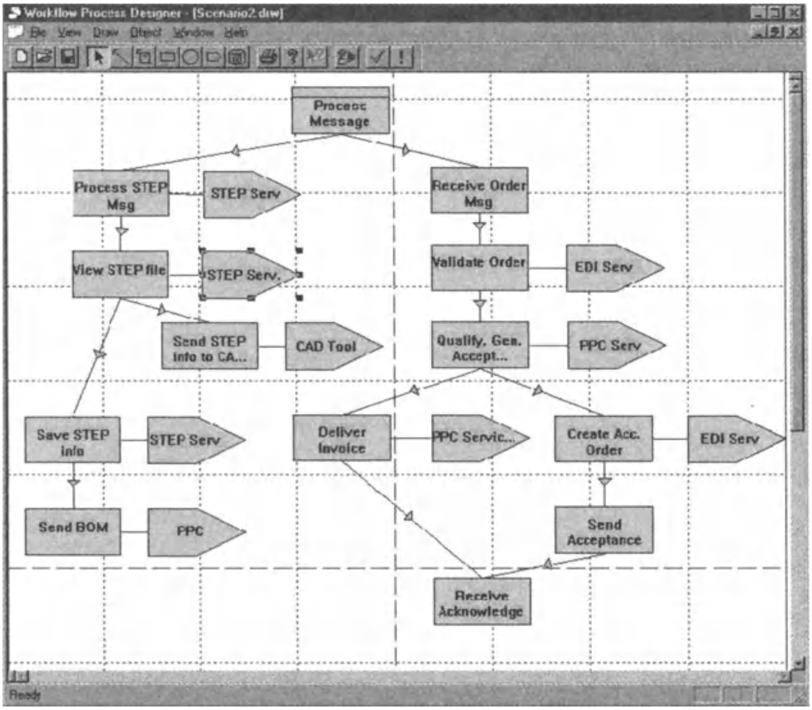


Fig. 18.24. Example of graphical workflow definition

representation is given by the Process Interchange Format (PIF) initiative[29]. In general, these tools are based on graphical interfaces / graphical languages and allow for business process modeling, structure and communication modeling, simulation, and analysis: cost, information usage, performance (cycle times, errors), detection of bottlenecks, resource utilization, etc.

Apart from the different graphical look, the underlying mechanisms in these tools are inspired by the IDEF0, workflow, or Petri nets developments. However, the main utilization of these tools is for modeling, simulation and analysis, not for the coordination itself and, so far, not easily usable for enterprise's activities coordination.

Coordination in MAS. The area of Multi-Agent Systems (MAS) (especially when involving Intelligent or Autonomous Agents) has also addressed the coordination issues and supporting mechanisms (Durfee et al. , 1987).

The interaction capability (both among different agents and between the agents and their environment) is one of the basic characteristics of an agent. Definition of high-level protocols – Agent Communication Languages (ACL) - to support these interactions has been a major work in the MAS area. KQML

[29] http://ccs.mit.edu/pif2.html

(Finin et al. , 1993) is one of the most widely known examples of such protocols. Another important mechanism resulted from the early works on MAS are the Contract Net and negotiation protocols (Smith , 1980) that can be used for task assignment.
Applications of these initial concepts have been developed in the area of scheduling and agile scheduling (Rabelo and Camarinha-Matos , 1994).
Coordination languages Development of coordination languages constitutes a new field of study in programming and software engineering, particularly motivated by the needs of the massively parallel and distributed systems. The starting point is the recognition of the need of programming models that explicitly deal with the concurrency of cooperation among large numbers of entities in massively parallel systems. Examples of such languages are LINDA, MANIFOLD, GAMMA, COOLL, PCL, etc [30]. For a comparative survey of coordination languages see (Papadopoulos et al. , 1998). In spite of this growing activity, none of these languages has yet entered the enterprise modeling area. One possible reason is that these languages have a complex syntax, more oriented to programmers / software developers than to business process modelers. The lack of simple graphical tools (although some initiatives are starting in this area, such as Visifold (Bouvry and Arbab , 1996)) is another disadvantage of these languages when compared to alternative solutions found in the business process modeling area.
An interesting idea, typically present in the workflow management systems (Lawrence , 1997), and more recently being also proposed by other works on coordination languages (Papadopoulos et al. , 1998) is the separation between *coordination* and *computing* or *service processing.* The computing part comprises the application-oriented functionality and the coordination part covers the communication and management of dependencies among activities (sequencing, synchronization, conditioned transitions, etc.). This general schema contributes to achieve a desired flexibility, since the data processing functions (the more stable part) are organized as libraries of services, whose invocation is managed by a separate coordination plan (Camarinha-Matos and Lima , 2001d, 1999b). From the coordination point of view, services are seen as black boxes with well-defined interfaces. Thus, changes in behavior are implemented by changing the coordination part only.
Finally, it shall be noted that a part of systems integration with major impact on users is the uniformity of the user interface. If the original (and different to each other) user interfaces of the various legacy components are kept, the user is not given a perception of using an integrated system. Therefore, "recovering" a legacy system to be used in an integrated environment implies the following (Fig. 18.25):

- Discard its "control structure" and replace with a global coordination system;

[30] references to these languages are available on the World-Wide Web

- Discard its (old) user interface and replace with by a uniform enterprise-wide user interface;
- Recover the basic functionalities of the legacy system, i.e. the basic processing services offered by this application and make them available as a library of services that can be invoked by the coordination system.

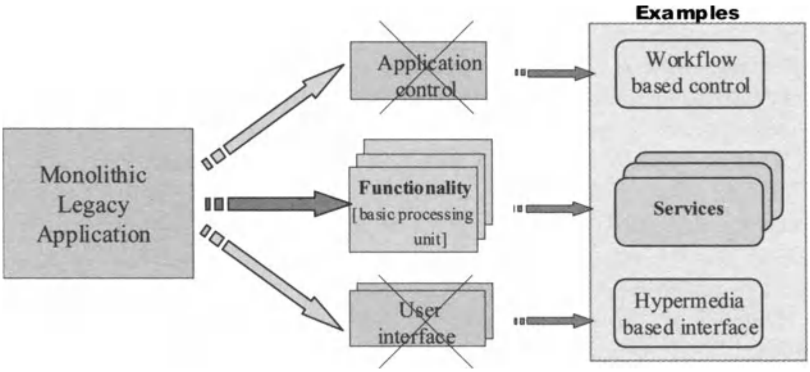


Fig. 18.25. Recovering the service components of legacy systems

Note: some legacy applications do provide 'batch-mode' functionality, which may be invoked from e.g. a command-line, a batch file or a macro commands file. In such cases it may be possible to operate such legacy applications directly from a newly designed user interface, without involving substantial reconstruction work. Since such 'batch-mode' functionality is usually quite limited, this approach may only be suitable as a temporary solution, e.g. so as to facilitate the migration to a fully converted system.

18.4.5 Integrated User Interfaces Design

As mentioned above, the adoption of a common user interface, or at least a common user interface paradigm, is a very important step in enterprise integration.

An approach that is becoming more and more popular is the adoption of a 'hypermedia' style of user interface, featuring the characteristics that can be explored by a web browser. Complementarily, and in the same direction, there is a trend to "open" the enterprise to the outside, offering access to its services, namely to the company's own employees and even to its clients and suppliers.

In order to support this approach, it is necessary to depart from the simple HTML pages that basically offer only static information (as shown in Fig.

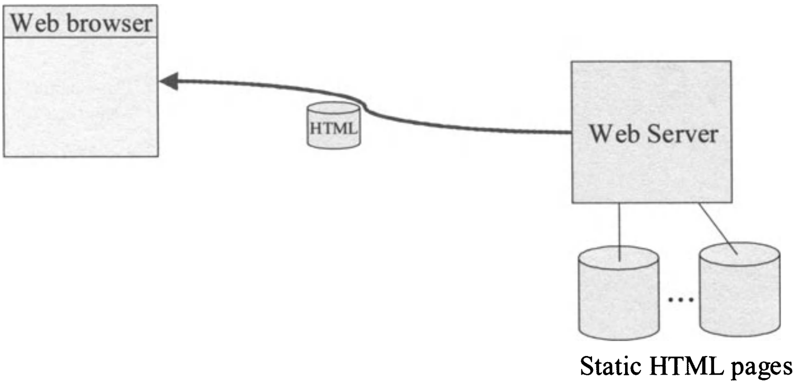


Fig. 18.26. Static HTML pages access

18.26), and move to dynamic pages generated 'on-the-fly' and whose contents are fed by the enterprise applications / subsystems.
One first mechanism that can be used to provide minimal dynamism is the CGI (Common Gateway Interface) that supports a limited bridge between a web server and enterprise applications (refer Fig. 18.27).

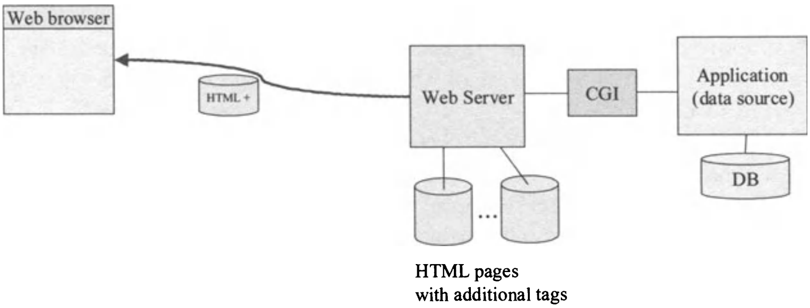


Fig. 18.27. Use of CGI for dynamic pages access

This mechanism is limited however and not easy to use. A more recent technology is offered by the so-called dynamic page servers (e.g. JSP, ASP) (as shown in Fig. 18.28).
These systems hide the details of the bridge between the server and the application, offering a more high level Application Programming Interface (API) that makes page development and deployment easier and faster.

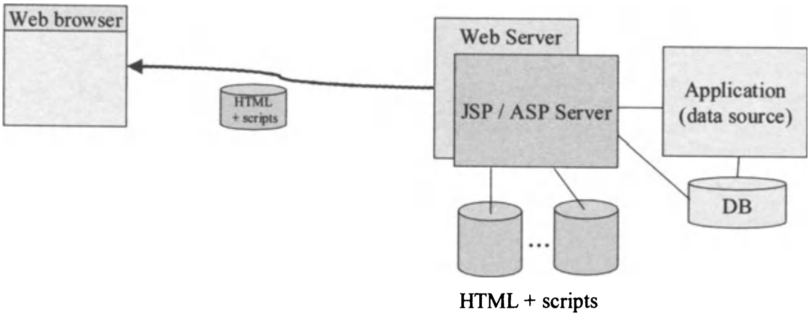


Fig. 18.28. Use of dynamic page servers

18.5 Inter-enterprise Infrastrcture Design

18.5.1 Introduction

Unlike in the past, most of the business processes are presently performed by several enterprises . Companies feel the need to focus on their core competencies and join efforts with others, in order to accomplish the requirements of the new products / services demanded by the market. In a cooperative networked organization, every enterprise is just a node that adds some value to the process - a step in the manufacturing / supply chain. Starting with the "outsourcing" wave of the 1980s, new collaborative network forms have emerged, leading to the concept of Virtual Enterprise (VE) and Virtual Organization (VO).

A VE / VO, being defined as a *temporary* consortium of enterprises / organizations that strategically join skills and resources, supported by computer *networks* in order to better respond to a business opportunity, embeds an implicit notion of agility. But in order to leverage the potential benefits of the VE / VO paradigm, flexible and generic **infrastructures** need to be designed and implemented (Camarinha-Matos and Afsarmanesh , 1999a; Bernus et al., 2002).

As a general requirement for an infrastructure to support a virtual organization, the involved organizations must be able to inter-operate and exchange a variety of information on-line, so that they can work as a single integrated unit with some common goals, while preserving their independence and autonomy. It is also necessary to point out that legacy systems running at present enterprises were not designed with the idea of directly connecting to corresponding systems in other enterprises. A typical situation is the case where enterprises exist individually before they decide to join in a cooperation network that requires both their sharing and exchange of information and their taking part in common tasks fulfillment. Consequently, every enterprise is autonomous, developed independently of other enterprises, and uses distinct approaches for

processing its local business information management, control, and decision making strategies that best serve its purposes.
Considerable investments have been made, namely in Europe and the USA, in a large number of research projects applying new organizational forms, which have produced an abundant variety of specific solutions and broad awareness for the necessary organizational changes. However, the research in many of these cases is highly fragmented, namely each project is focused on solving specific problems and applying IT to partially design and develop a minimal business-to-business interaction mechanism to support its basic needs. As such, there is no effective consolidation / harmonization among these research projects in order for them to have an effective impact.
The availability of a large number of contributing elements (such as technologies, paradigms, best practices and models) constitutes a set of enabling factors for the opportunity of materializing concepts that were waiting for such enabling factors. However, most of these technologies and concepts are in their infancy and under development, mostly fragmented and non-interoperable. Therefore, the design of an infrastructure for inter-enterprise collaboration still faces a number of difficulties:

- The implementation and configuration of a VO support infrastructure currently requires considerable engineering effort, an obstacle to dynamic VOs;
- Even the most advanced infrastructures coming out of leading R&D projects require complex configuration and customization processes hardly manageable by small and medium enterprises (SMEs);
- Almost all VO / VE projects are forced to create their own infrastructure, due to the lack of common reference architectures (Tolle et al, , 2002) and interoperability principles. This represents a deviation of some resources of the project from the main focus, while generating something only applicable to each specific project;
- Only a few projects if any, address the development of a comprehensive horizontal infrastructure;
- There is a need to discern between enabling vs. disabling technologies. Some infrastructure development efforts are too biased by short-term technologies, which might represent an obstacle for their adaptation by non-ICT SMEs.

Furthermore, the integrating infrastructure should support all phases of the VE / VO life cycle (Fig. 18.29), but most developments so far have concentrated only on the creation and operation phases.
In other words, there is a need for a base (horizontal) infrastructure that supports basic information sharing and exchange and basic coordination, developed based on a safe communications infrastructure, but also a number of specific functions to support various phases of the VE / VO life cycle.

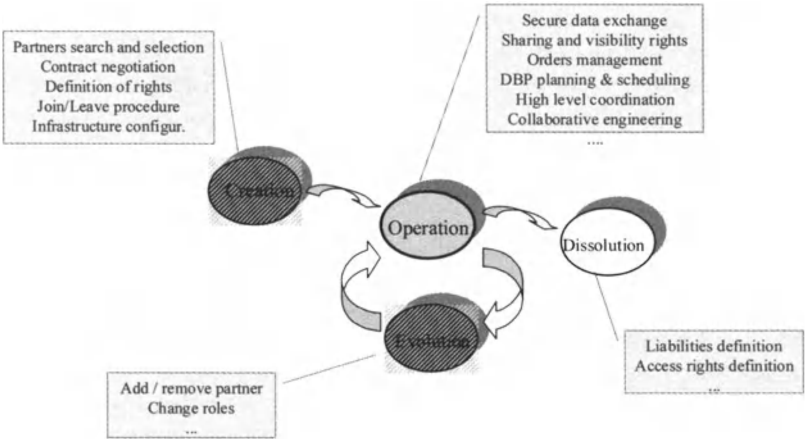


Fig. 18.29. VE / VO life cycle and its needs

18.5.2 Main Approaches

Three main approaches have been followed by the designer of new VO / VE infrastructures including layer-based, agent-based, and service federation based frameworks.
Layer-based (transactional) frameworks add a cooperation layer to the existing ICT platforms of enterprises. Inter-enterprise cooperation is then performed via the interaction (transaction-oriented) through these layers. These frameworks include client-server solutions, web-enabled servers, and frequently adopt workflow models and standards for information exchange (STEP, EDIFACT, XML, etc.). Figure 18.30 shows the PRODNET[31] infrastructure (Camarinha-Matos and Afsarmanesh , 1999a), a typical example of a layer-based framework.
The main components of this infrastructure (PRODNET Cooperation Layer) are (Camarinha-Matos et al. , 2001c):

- PRODNET Communications Infrastructure (PCI), which is responsible for handling all communications with other nodes in the network. It includes functionalities such as selection of communication protocols and channels, basic communications management, privacy mechanisms, digital signatures, and secure message communication channels between nodes.
- Distributed Information Management System (DIMS), which is responsible to model and manage the exchange of all integrated cooperation-related information, while preserving the autonomy and information privacy of involved enterprises. A federated information management ap-

[31] PRODNET: platform for production planning and management in virtual enterprises

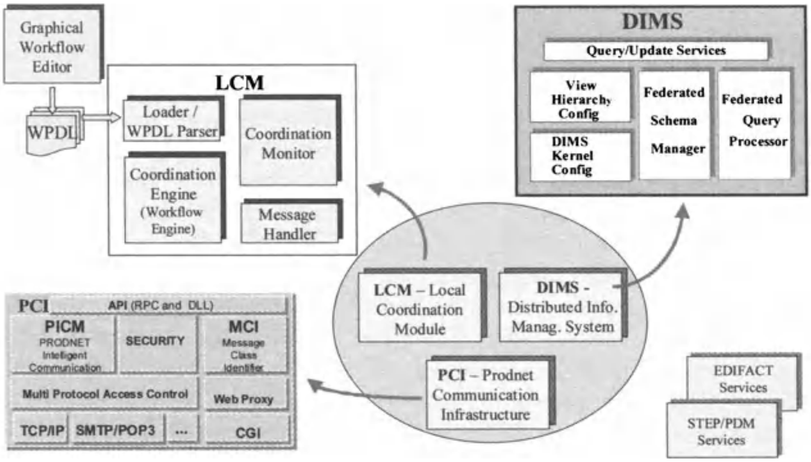


Fig. 18.30. Main components of the PRODNET cooperation layer

proach is adopted in PRODNET as a way to preserve the nodes' autonomy while providing transparent access to the shareable information (Afsarmanesh et al., 1999).

- Local Coordination Module (LCM), which handles all cooperation events according to a set of rules which are specified for each particular enterprise (Camarinha-Matos and Lima , 2001d). This component of PRODNET acts as a workflow engine, according to the reference model of the Workflow Management Coalition and is the "executor/controller" of the activity flow plans defined by a graphical Configuration Component. Namely, it is responsible for the behavior of the PCL internal modules' interactions. These cooperation events have an asynchronous nature and are invoked by other nodes of the VE, by the internal modules of the enterprise, or by other modules of the cooperation layer.

A set of additional services were also developed: Electronic Data Interchange (EDI) module for preparation and receiving of orders and other business-related messages in EDIFACT format, and a STEP module to handle the technical product data used within the VE.
In addition to these base components, the PRODNET infrastructure includes services to support partners search and selection, monitor and manage distributed business processes, configure the platform, and interact with the PPC / ERP systems.
Agent-based frameworks represent enterprises as agents and the inter-enterprise cooperation as interactions in a distributed multi-agent system.
A growing number of works are being published on the application of multi-agent systems and market-oriented negotiation mechanisms for the VE / VO formation (Camarinha-Matos and Afsarmanesh , 2001a). One such example

can be found in Rocha and Oliveira (1999). This work assumes a ***virtual market place*** where enterprises, represented by agents that are geographically distributed and possibly not known in advance, can meet each other and cooperate in order to achieve a common business goal. A MAS architecture is proposed to model the electronic market to support the formation of the VE. In addition to the agents representing the enterprises, there is a market agent – coordinator or broker – that is created and introduced in the MAS community when a business opportunity is found. A multi-round contract-net protocol is followed: the market agent sends invitations to the electronic market corresponding to each of the VE sub-goals; the marker agent then receives bids and evaluates them. The most favorable bids are selected, based on a multi-criteria mechanism and constraint-based negotiation. Examples of considered criteria are lower-cost, higher quality, higher availability, etc. Multiple negotiation rounds can take place. At the end of each round, bidders are notified whether their bids are wining or loosing and a rough qualitative justification is provided, allowing them to change the parameters of their proposals.

A similar work is found in Li et al. (2000) where a more detailed analysis of the problem of goal decomposition (leading to a hierarchy of VE goals) is suggested. In addition to the enterprise agents and VE coordinator agent (broker), an information server agent is introduced in order to keep public information related to common organizational and operational rules, market environment, enterprises and products / services provided, etc. The need for a common ontology to support the communication among agents is explicitly introduced and a multi-attribute, constraint-based negotiation / selection process is implemented.

The work described in Shen and Norrie (1998) identifies the need for 'Yellow Pages' agents that are responsible to accept messages for registering services (similar to the information agent server mentioned above). They also consider the concept of Local Area, a quasi-physical division of the network that can be controlled by a local area coordinator. This is a similar concept to the Local Spreading center first introduced by the HOLOS system (Rabelo et al. , 1999). These proposals are however limited by a number of factors which affect their practical implementation including:

- lack of common standards and ontologies, a situation difficult to overcome in a general "open universe" of enterprises;
- none of these proposals takes into account more subjective facets like trust, commitment, successful cooperation history, etc;
- in general, they pay little attention to the implantation aspects and the management of the yellow pages / market place;
- security issues in the negotiation process are not addressed; a critical point is that agents - representing enterprises - are only partially cooperative. They might be self-interested, competitive, and even exhibit antagonistic behaviour.

The attempt to reach a fully automated decision-making process, although an interesting academic exercise, is quite unrealistic in this application domain. Understanding the VE / VO formation process, modeling it and developing support tools, are open challenges. Initiatives such as (UDDI[32]) or WSDL[33] need to be considered in more practical MAS approaches.
Regarding the operation phase, many approaches assume that simple mechanisms of inter-agent cooperation are sufficient. One example is given by Mobile Intelligent Agents for Managing the Information Infrastructure (MIAMI) (Broos et al., 2000) that developed a mobile agents platform for VEs supporting a virtual marketplace for VE creation and a monitoring service used during the operation of the VE in order to supervise processes and provide to partners (having the appropriate permissions) the information about the state of partial processes. Global coordination is supported by a workflow management system. Another example is MetaMorph II (Shen , 2001), which developed a mediator-based multi-agent architecture to support enterprise integration and supply chain management.
Service-federation / service-market frameworks - According to this model, enterprises should be able to plug/unplug their services to/from service directories. By means of proper "standard" service interfaces, the interoperability with other (remote service requesting) enterprises is supported, regardless of the heterogeneity associated with the actual implementation of these services.
The JINI architecture (Sun , 1999) for instance, can support (in a transparent way) a federation of service functions offered by different service suppliers and running on different nodes of a network. Special directories (called JINI Lookup Services) provide the registration of service specifications and the associated functions to search such services, based on a configurable search criterion. According to this model, and in order to plug/unplug their services to/from the service directories, enterprises must agree on a proper "standard" service interface.
Although JINI illustrates an implementation model for service federation, its use in VE / VO environments requires its adaptation to operate in Wide Area Networks and the development of additional support functionalities. For instance, in VEs the subject of access rights and enterprises information visibility is very important, and therefore when searching and accessing a specific service type it is important to determine both the supplier of the service and the requesting client, in order to check for authorization and visibility rights. It is also necessary to define the rules for both service specification/definition and service registration through the service interface. Furthermore, the general acceptance of the service interfaces by service providers is of great importance, since service providers must in fact develop their services or at least wrap them to be compliant with such rules.

[32] Universal Description, Discovery and Integration (see http://www.uddi.org)
[33] Web Service Definition Language

It shall be noted that the three mentioned frameworks are not necessarily exclusive; therefore, mixed approaches can be found. For instance, services offered in a distributed environment can be accessed by remote invocation (via proxy objects) or by mobile agents. Although in principle all these approaches provide the base mechanisms necessary for the integration of enterprises from any sector, it is important to note that different organizational forms from different domains of application may fit substantially better if implemented following a specific approach from the three frameworks presented. For instance, most manufacturing applications may fit better with in layered infrastructure, while applications in service provision sector can better benefit from a service federation infrastructure. Similarly, the tele-monitoring and tele-supervision, if requiring high levels of autonomy, fit the agent-based infrastructure better. In other words, once again, the choice of a best methodology for every system starts with a careful analysis of the system itself, in order to identify all its specificities and characteristics before deciding on the best approach and tools to choose.

On top of the base horizontal infrastructure, there is a need to support the definition and execution of the collaborative business processes. When a Business Process (BP) is executed by a Virtual Enterprise, parts of the decomposition of this BP (i.e. sub-processes) are assigned to different enterprises, making it a ***Distributed Business Process*** (DBP). Coordination of DBPs and activities is an important element in the VO operation. Once a DBP is defined, scheduled tasks and responsibilities are assigned to each individual partner. The successful achievement of the common goal (i.e. the delivery of the final product or service to the client)depends on the proper and timely operation of each VO / VE member (and each supporting service in each VO / VE member). Most of the early approaches to DBP modeling and coordination took a workflow-based approach, starting with the WfMC reference architecture (WfMC, 1994) and experimenting with extensions for supervision of distributed processes (cross-organizational workflow), including some preliminary but very limited works on exception handling, multi-level coordination and its relationship to coordination roles of the VO members and flexible workflow models to support less structured processes. In addition, a number of multi-agent based approaches using the negotiation / contract net protocols have been suggested in order to handle dynamic task allocation and re-allocation.

In terms of DBP planning and modeling some graphical languages have been suggested but a standard is still necessary in order to allow effective distribution / sharing of business processes. Proposals like the Process Specification Language (PSL) (Schlenoff et al. , 2000) have been discussed but there is still more work to do. A recent European initiative, the Unified Enterprise Modeling Language (UEML) network, might contribute to some harmonization for this modeling area.

In terms of other business support functions, most of the past projects have addressed application-specific cases. ERP vendors have been extending their

monolithic enterprise-centric systems in order to comply with more dynamic supply chains and networks, but this is an area requiring more investment on generic functionalities.
Some important design challenges in inter-enterprise collaboration support infrastructures include (Camarinha-Matos , 2002a,c):

- Establishment of common reference models for VE / VO (Tolle et al, , 2002);
- Development of generic, interoperable, progressively invisible and safe support infrastructures;
- Development of new generic business support functions for collaborative processes;
- Coordination, administration and management of highly distributed activities, tasks, processes, and roles;
- Advanced federated information management, supporting the authorized information exchange, information integration mechanisms, and collaborative work among organizations;
- Design, development, provision and management of value-added services provided in the context of VE / VOs;
- Management and configuration of local/regional clusters operating on specific vertical markets with specific business processes requirements;
- Tools for dynamic evaluation of revenues, rights and liabilities of every VE / VO relationships, and dynamic selection of new partners in order to make the VE network more effective;
- Soft-modeling and reasoning with special decision support mechanisms for supply chain management, selection of partners or efficiency of VE / VO relationships;
- Comprehensive VE / VO creation and dissolution frameworks;
- E-notary services for networked organizations with special services such as: black lists, past performance, credentials, best practices;
- Legal framework and e-contract management and adaptation to the legal frameworks;
- Inter-domain transactions, business processes sharing and management;
- Effective integration of legacy systems as well as methodologies for transforming existing organizations into VE / VO-ready organizations;
- Tracking and auditing support services;
- Advanced simulation models and tools for networked collaborative organizations;
- Mechanisms for handling post-cooperation property rights and liabilities;

The development of such functionalities cannot be separated from the socio-organizational issues. Therefore, a stronger emphasis on multidisciplinary research is necessary in this field.

18.5.3 Breeding Environments

The concept of cluster of enterprises (which should not be confused with a VE) represents a group or pool of enterprises and related supporting institutions that have both the potential and the will to cooperate with each other through the establishment of a long-term cooperation agreement(according to Camarinha-Matos and Afsarmanesh (2001b)). Buyer-supplier relationships, common technologies, common markets or distribution channels, common resources, or even common labor pools are elements that typically bind the cluster together. This is not a new concept, as a large number of related initiatives have emerged during the last decades, such as in Europe and USA, or internationally[34]. But the advances in information and communication technologies now bring new opportunities to leverage the potential of this concept, namely by providing the adequate environment for the rapid formation of agile virtual enterprises. In fact, although not well understood earlier, it is now clear that the formation of dynamic VO / VE requires an appropriate breeding or "nesting" environment in order to guarantee basic requirements such as:

- trust building - trusting your to work with other enterprises is a long-term process;
- common infrastructures and agreed upon business practices - requiring substantial engineering / re-engineering effort;
- a sense of community;
- a sense of stability;
- etc.

A cluster represents a long-term organization and therefore presents an adequate environment for the establishment of cooperation agreements, common infrastructures, common ontologies, and mutual trust, which are the facilitating elements when building a new VE.
For each business opportunity found by one of the cluster members, a subset of the cluster enterprises may be chosen to form a VE for that specific business opportunity. In this perspective, the expected competitive advantage of cooperative development of products and services creates a tie among the cluster members. The more frequent situation is the case in which the cluster is formed by organizations located in a common region, although geography is not a major facet when cooperation is supported by computer networks. Nevertheless, the geographical closeness has some advantages for cooperation as it may facilitate better adaptation to the local (culture) needs and an easier creation of a "sense of community".

[34] e.g. the Globemen (GLOBal Engineering and Manufacturing in Enterprise Networks) consortium which regards such clusters of enterprises as Networks which further create Virtual Enterprises (and Reference Models of such VEs) as needed for a particular set of customer needs.

In terms of support infrastructure, the cluster enterprises are normally "registered" in a directory, where their core competencies are "declared". Based on this information, the VE initiator / creator (**broker**), which is usually a member of the cluster enterprises, can select partners when a new business opportunity is detected. Clearly, several VEs can co-exist at the same time within a cluster, even with some members in common.
The cluster does not need to be a closed organization; new members can adhere but they have to comply with the general operating principles of the cluster. When forming a VE, preference will be given to the cluster members but it may become necessary to find an external partner in case some skills or capacities are not available in the cluster. The external partner will naturally have to adjust and accept the common infrastructure and cooperation principles. In addition to enterprises, a cluster might include other organizations (such as research organizations, sector associations, etc.) and even free-lancer workers. The establishment and management of clusters through adequate infrastructures represent therefore an important support for the creation of agile virtual enterprises (Fig. 18.31).

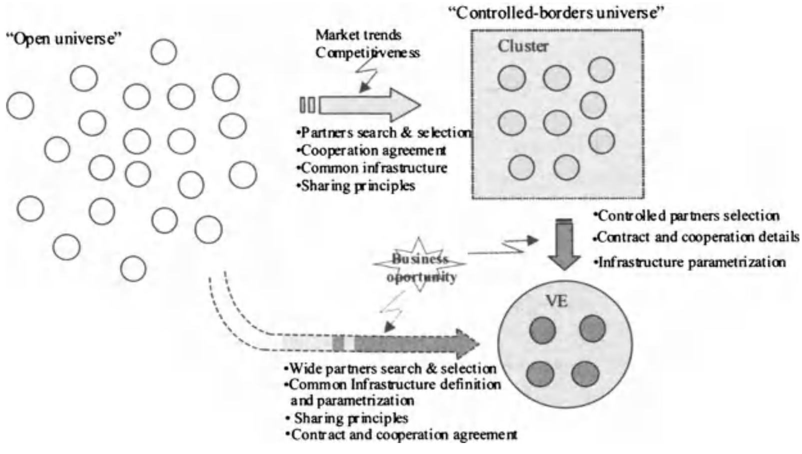


Fig. 18.31. - Two approaches for VE formation

The idea of using a cluster as the basis for the formation of virtual enterprises has been identified in other research works such as COSME/VIRPLAS (Molina et al. , 1998) or VIRTEC (Bremer et al., 1999) projects. These projects have identified some of the major characteristics and needs of cluster management, but did not introduce the necessary IT infrastructure and support tools. MASSYVE is an example of a project that adopted a multi-agent approach for the management of industry cluster and support of a broker for VE creation (Rabelo et al. , 2000). Some of the aspects that have been subject of

more attention in recent projects regarding cluster management are skill definition, brokerage, partners search and selection ('e-procurement'), definition of cooperation rules (negotiation and contracting), etc. The concept of cluster is evolving in parallel with the emergence of other forms of relationships, such as 'communities of practice' or 'virtual communities', representing a general trend to the emergence of a kind of 'society of relationships'.
Some important challenges faced in the design of a cluster management system (Camarinha-Matos , 2002a,c) are:

- Management and configuration of local/regional clusters operating on specific vertical markets with specific business processes requirements, including appropriate skills specification (e.g. using WSDL);
- The full VO / VE creation framework;
- E-notary and certification services for networked organizations with special services such as black lists, past performance, credentials, and best practices;
- Modeling and management of cooperation contracts and agreements;
- Methodologies for transforming existing organizations into VO-ready organizations.

18.5.4 Virtual Communities Support

When a proper cooperation nesting environment (e.g. a cluster or long term network) is in place, **Virtual Communities of Practice (VCP) / virtual teams** emerge within such an environment, constituting a fundamental element of value creation and sustainability for enterprises. Virtual Communities and Communities of Practice are not new concepts, but they acquire specific characteristics and increased importance when considered in the context of the collaborative networks of organizations. These communities, although spontaneously created, are bound to certain social rules resulting from the commitment of their members to the underlying organizations (new concept of social-bound VCPs).
This is the case, for instance, in *concurrent* or *collaborative engineering* where teams of engineers, possibly located in different enterprises, cooperate in a joint project such as for example the co-design of a new product (Bremer et al., 1999; Putnik and Pithon , 2002). A large number of computer supported cooperative tools are becoming widely available for synchronous cooperation. Some examples are teleconference and chat tools combined with application sharing mechanisms. Considering the geographical distribution, the autonomy of the VE members, the local corporate cultures and also the individual working preferences of the team members, it is likely that most of the activities will be carried out in an asynchronous way, which raises important questions in terms of coordination.
For this purpose, several approaches to develop *flexible workflow* systems have been proposed. In the case of processes mainly executed by humans, rigid

forms of procedural control are not adequate. People like to keep their freedom regarding the way they work. Product design, like any other creative process, evolves according to a kind of "arbitrary" flow. It is therefore necessary to also support *loosely constrained* sets of *business processes.*
The trend is followed by other communities of professionals (e.g. consultants) that share the body of knowledge of their professions such as similar working cultures, problem perceptions, problem-solving techniques, professional values and behavior.
Another special area of application, as for instance represented by the TeleCARE project (Camarinha-Matos , 2002b), is the creation of virtual communities for providing care services to elderly. These communities involve organizations and people such as care providers, health care professionals, relatives of elderly, and the elderly. In addition to a base infrastructure providing secure communications and the information and resources sharing, these communities require specific tools or vertical services such as elderly life status monitoring, agenda management, entertainment, time bank management, etc.
In general, the design and development of an advanced infrastructure for virtual communities requires (Camarinha-Matos , 2002a,c):

- Development and management of shared, smart spaces for geographically distributed teams of the same or different organizations, which develop complex engineering products;
- Provision of adequate visibility and access rights definition and management;
- Understanding and modeling multi-level relationships among VCP members;
- Coordination of (asynchronous) activities performed in different places by different actors; flexible coordination models;
- Frameworks for collaboration in mobile contexts;
- Tools for community management, leadership, and creation of incentives;
- Provision of notification mechanisms regarding major events in the design / planning process (e.g. conclusion of a step by an actor);
- Mechanisms to handle Intellectual Property in VCP under social contracts;
- etc.

18.5.5 GRID Infrastructure for Enterprise Networks

A new *entry* in the infrastructures design that may have substantial impact on infrastructures for inter-enterprise cooperation is **Grid computing** (Foster et al. , 2001). In the late 90s, Grid computing was proposed first only as a distributed computing infrastructure for advanced science and engineering. In scientific application domains e.g. physics, astronomy, biology, etc., a major requirement is the sharing of very complex and expensive resources, while another important requirement is to achieve high-performance.

Grid computing enables coordinated and dynamic resource sharing to support collaborative problem solving in a large scale distributed computing environments (Afsarmanesh et al., 2001). It provides direct access to hardware, software, and data, through a set of uniform interfaces.
Currently, the Globus toolkit (Foster et al. , 2002) is a reference implementation of the Grid concepts, and is the *de facto* standard for today's Grid computing. The Globus toolkit employs a "bag of services" approach. Users can select the set of tools that they need for their specific requirements during application development. In this approach, the "bag" can also be extended with additional tools. The toolkit includes resource management services, such as resource discovery (using LDAP based catalogues), resource allocation and monitoring. There are also some services for efficient and secure data transfer (GridFTP), data replication and replica management. The Globus toolkit also implements the Grid Security Infrastructure (GSI) defined by the Grid computing, which describes techniques for authentication in wide area networks.
Considering the required data and information management services to be provided by a Grid, several important issues need to be mentioned:
Efficient low-level data exchange and communication services are provided in Grid, with standard data repository operations (e.g. file transfer, move, replicate, extract). There is also provision for processes to run specific tasks at specific destinations. Therefore tele-supervision requiring mobile agents can be implemented using Grid technology. The Grid infrastructure also provides information catalogs for resources.
A number of services are currently being developed as part of various research projects, e.g. services for accessing relational databases, uniform access to relational data sources from within the Grid environment, the Grid security infrastructure, and some mechanisms for data transformation (from one data type into another, mostly specific data files obtained from an instrument).
However, semantic integration of data has not been addressed yet. Main focus is currently on integrating database access within Grid and using existing Grid services as much as possible.
Grid technology provides a strong base framework for the distributed computing and resource sharing, which can be used to implement high-performance data integration for scientific collaboration. However, the research is not yet mature, and many higher-level services are needed to fully support VOs. The following can be mentioned as some of the requirements for data/information integration to support VOs:

- Standard information management capabilities are required: DDL/DML[35] language (e.g. SQL-like), for common meta-data definition and information manipulation;
- The best path for exchange of large data sets among VO members can be identified based on the on-line monitoring of distributed resources;

[35] Data Definition Language / Data Modeling Language

- Public directories of enterprise profiles, competencies, and capabilities can be implemented on top of the Grid information catalogues. Lightweight Directory Access Protocols (LDAP) can be applied;
- Since the Grid infrastructure relies on the existing operating systems and/or resource managers, higher-level management tools are needed to configure and maintain the resources available through a VO;
- Distributed service management functionalities are required to be supported, namely for service definition (meta-data), plug/unplug service implementations (proxies) and service discovery facilities. The current Grid computing research is also directed towards the (Web) services support;
- Supporting node autonomy/privacy is required. Facilities for dynamic definition of information (and services) access rights and provision of different visibility levels for different VO members are needed;
- Information privacy, fine-grained visibility levels, and strong communication security must be enforced.

Grid enables the execution of processes on specified remote hosts. This technology can be used to implement data collection/integration 'agents', which can be used as the base for distributed/federated query processing.
The Grid technology, though still mostly in design stage or partially developed, is receiving world-wide attention and a lot of public investments. Therefore, the need for such technology is already well established and there is a strong chance that Grid will be accepted as the standard for a pervasive communications infrastructure.

18.6 Conclusion

Information technology (IT), as a subsystem of the Information System is a vital component of any modern organization, which due to the wide scope of necessary functionalities cannot be built around the tools provided by a single vendor. Therefore, designing an enterprise IT subsystem is mainly a problem of integrating a large variety of legacy systems.
On the other hand, component-based systems rely on a multitude of technologies that face a rapid evolution and versioning. As a consequence, there is no simple solution for enterprise systems integration, and certainly no off-the-shelf solution. Systems integration (and consequently systems interoperability) is a continuous challenge and need. This problem can be handled at various levels, namely for manufacturing at cell / shop-floor, intra-enterprise, and inter-enterprise levels.
For each level, different technologies have been offered in each historic phase and the choice / design of solutions depends to some extent on the specificity of these technologies. Nevertheless, some general steps for systems integration can be identified, e.g.: adoption of reference models, systems modeling, harmonization of components, information and knowledge inter-linking, and design of integrated coordination models.

With the emergence of new organizational forms, namely new collaborative networks, and in parallel the evolution of computer networking technologies, in addition to the global problem of interoperability, a wide range of new research challenges have emerged requiring new approaches and considerable research and development efforts.

References

Afsarmanesh, H., Wiedijk, M.and Hertzberger, L. O. - Flexible and Dynamic Integration of Multiple Information Bases, Proceedings of the 5th IEEE Inter. Confer. on "Database and Expert Systems Applications DEXA'94", Athens, Greece, Lecture Notes in Computer Science (LNCS) 856, Springer Verlag, Sep. 1994.

Afsarmanesh, H., Garita, C., Ugur, Y., Frenkel, A., and Hertzberger, L. O. - Design of the DIMS Architecture in PRODNET, in Infrastructures for Virtual Enterprises - Networking Industrial Enterprises, Kluwer Academic Publishers, ISBN 0-7923-8639-6, Oct 1999.

Afsarmanesh, H.,Kaletas, E. C. and Hertzberger, L. O. – A reference architecture for scientific virtual laboratories, Future Generation Computer Systems, vol. 17, 2001.

AMICE – CIM-OSA: Open Systems Architecture for CIM, 2^{nd} revised and extended version, Springer-Verlag, Berlin, 1993.

Arbab, F. – What do you mean, Coordination?, Bulletin of the Dutch Association for Theoretical Computer Science, Mar 1998.

Barata, J. and Camarinha-Matos, L.M. – Contract management in agile manufacturing systems, in Collaborative Business Ecosystems and Virtual Enterprises, Kluwer Academic Publishers, May 2002.

Barata, J.and Camarinha-Matos, L.M. – Implementing a contract-based multi-agent approach for shop floor agility, in Proceedings of DEXA 2002 – 13th Int. Workshop on Database and Expert Systems Applications / HoloMAS – 3rd Int. Workshop on Industrial Applications of Holonic and Multi-Agent Systems, Aix-en-Provence, France, 2-6 Sep 2002, IEEE Computer Society, ISBN 0-7695-1668-8, pp. 639-645.

Basseto, S., Siadat, A. and Martin, P. – Introduction to the prototyping of intelligent supervision systems, in Proceedings of BASYS'02 – Knowledge and Technology Integration in production and services, Kluwer Academic Publishers, ISBN1-4020-7211-2, 2002.

Bernus, P., Baltrusch, R., Vesterager, J. and Tølle, M. - Fast tracking ICT infrastructure requirements and design, based on enterprise reference architecture and matching reference models, in Proceedings of PRO-VE 2002 – Collaborative Business Ecosystems and Virtual Enterprises (L.M. Camarinha-Matos, ed.), Kluwer Academic Publishers, ISBN 1-4020-7020-9, Sesimbra, Portugal, 1-3 May 2002.

Bellifemine, F., Poggi, A. and Rimassi, G., "JADE: A FIPA-Compliant agent framework", Proc. Practical Applications of Intelligent Agents and Multi-Agents, April 1999, pg 97-108

Bond, A. and Gasser, L. - An Analysis of Problems and Research in DAI, Readings in Distributed Artificial Intelligence, pp.3-35, Morgan Kaufmann, 1988.

Bouvry, P., and Arbab, F., (1996) Visifold: A Visual Environment for a Coordination Language. In P. Ciancarini and C. Hankin (Eds), Proc. COORDINATION '96, LNCS 1061Berlin : Springer, pp 403—406

Bremer, C. F., Mundim, A., Michilini, F., Siqueira, J.and Ortega, L., 1999 - A Brazilian case of VE coordination, in Infrastructures for Virtual Enterprises - Networking Industrial Enterprises, L.M. Camarinha-Matos, H. Afsarmanesh (Ed.s), Kluwer Academic Publishers, ISBN 0-7923-8639-6, Oct 1999.

Broos, R., Dillenseger, B., Guther, A.and Leith, M. – MIAMI: Mobile intelligent agents for managing the information infrastructure, www.infowin.org/acts/analysys/products/thematic/agents/ch3/miami.htm, 2000.

Camarinha-Matos, L.M. - Virtual organizations in manufacturing: Trends and challenges, (invited keynote), in *Proceedings of FAIM 2002, 12th Int. Conf. On Flexible Automation and Intelligent Manufacturing*, Publisher Oldenbourg (Munich), ISBN 3-486-27036-2, pp. 1036-1054, Dresden, Germany, 15-17 Jul 2002.

Camarinha-Matos, L.M.and Afsarmanesh, H. – Infrastructures for Virtual Enterprises - Networking Industrial Enterprises, Kluwer Academic Publishers, ISBN 0-7923-8639-6, Oct 1999.

Camarinha-Matos, L. M. and Afsarmanesh, H. - Virtual Enterprise Modeling and Support Infrastructures: Applying Multi-Agent Systems Approaches, in Multi-Agent Systems and Applications, M. Luck, V. Marik, O. Stpankova, R. Trappl (eds.), Lecture Notes in Artificial Intelligence LNAI 2086, Springer, July 2001.

Camarinha-Matos, L. M.and Afsarmanesh, H. - Service Federation in Virtual Organizations, Proceedings of PROLAMAT'01, 7-10 Nov 2001, Budapest, Hungary.

Camarinha-Matos, L. M.and Afsarmanesh, H. - Design of a virtual community infrastructure for elderly care, in Proceedings of PRO-VE 2002 – *Collaborative Business Ecosystems and Virtual Enterprises*, Kluwer Academic Publishers, ISBN 1-4020-7020-9, pp.439-450, Sesimbra, Portugal, 1-3 May 2002.

Camarinha-Matos, L. M.and Afsarmanesh, H. – Dynamic Virtual Organizations, or not so dynamic?, in Proceedings of BASYS'02 – Knowledge and Technology Integration in production and services, Kluwer Academic Publishers, ISBN1-4020-7211-2, 2002.

Camarinha-Matos, L. M.and Afsarmanesh, H.; Osorio, A.L. - Flexibility and safety in a web-base infrastructure for virtual enterprises, International

Journal of Computer Integrated Manufacturing (Taylor & Francis), Vol. 14, N. 1, Jan 2001, pp 66-82.

Camarinha-Matos, L.M., Barata, J.and Flores, L. - Shopfloor Integration and Multiagent based Supervision, Proceedings of INES'97 – IEEE Int. Conf. On Intelligent Engineering Systems, ISBN 0-7803-3627-5, pp 457-462, Budapest, Hungary, Set 1997.

Camarinha-Matos, L.M.and Lima, C. - Cooperation coordination in virtual enterprises, Journal of Intelligent Manufacturing (Kluwer), Vol. 12, N^o 2, ISSN 0956-5515, Apr 2001, pp. 133-150.

Camarinha-Matos, L.M.and Lima, C. - PRODNET coordination module, in Proceedings of PRO-VE'99 - IFIP Int. Conf. on Infrastructures for Virtual Enterprises (L.M. Camarinha-Matos, H. Afsarmanesh– Eds.), Kluwer Academic Publishers, ISBN 0-7923-8639-6, pp. 147-166, Oct 99, Porto, Portugal.

Camarinha-Matos, L.M.and Osorio, A.L. - Monitoring and Error Recovery in Assembly Tasks, 23rd ISATA - Int. Symp. on Automotive Technology and Automation, Vienna, Austria, Nov. 90.

Carriero, N.and Gelertner, D. – Coordination languages and their significance, Communications of the ACM, Vol. 35 (2), 1992, pp. 97-107.

CASA/SME – The new manufacturing enterprise wheel, 1993. http://www.sme.org/memb/casa/wheel2.html.

Colombo, A. W. – Development and implementation of hierarchical control structures of flexible production systems using high-level Petri nets, PhD Thesis, Meisenbach Verlag Bamberg, Mar 1998, ISBN 3-87525-109-1.

Durfee, E., Lesser, V.and Corkill, D. - Coherent Cooperation among Communication Problem Solvers, IEEE Transactions on Computers, N 36, pp.1275-1291, 1987.

Finin, T., Weber, J.and Wiederhold, G. – Specification of the KQML Agent-Communication Language, The DARPA Knowledge Initiative External Interfaces Working Group, June 15, 1993.

Foster, I., Kesselman, C.and Tuecke, S. – The Anatomy of the Grid – Enabling scalable virtual organizations, Int. J. Supercomputer Applications, 2001.

Foster, I., Kesselman, C., Nick, J. and Tuecke, S., Grid Services for Distributed System Integration. *Computer*, 35(6), 2002.

IFIP-IFAC Task Force - GERAM: Generalised Enterprise Reference Architecture and Methodology, Version 1.6.3 (March 1999), www.cit.gu.edu.au/~bernus/taskforce/geram/versions/geram1-6-3/v1.6.3.html

Integrated Manufacturing Technology Roadmapping Initiative – Technologies for Enterprise Integration: Chapter 4 - Enabling Infrastructure, Dec 1999.

Lawrence, P. – The Workflow Handbook 1997, Wiley, Oct 1997, ISBN 0-47196947-8.

Li, Y., Huang, B.Q., Liu, W. H., Wu, C.and Gou, H.M. – Multi-agent system for partner selection of virtual enterprises, Proceedings of 16th IFIP World Computer Congress 2000, Vol. ITBM, Beijing, China, 21-25 Aug 2000.

Lopes, L. S.and Camarinha-Matos, L.M. - Learning a Taxonomy of Failures in Assembly, in Making Robots Smarter – Combining sensing and action through robot learning, K. Morik, M. Kaiser, V. Klingspor (Eds.), Kluwer Academic Publishers, 1999, pp. 161-184, ISBN 0-7923-8562-4.

Malone, T. W.and Crowston, K. – The interdisciplinary study of coordination, ACM Computing Surveys, Vol. 26 (1), 87-119, March 1994.

Molina, A., Flores, M.and Caballero, D., 1998 – Virtual enterprises: A Mexican case study, in Intelligent Systems for Manufacturing, L.M. Camarinha-Matos et al. (Eds.), Kluwer Academic Publishers, ISBN 0-412-84670-5.

OAG – Open Applications Integration White Paper, Open applications Group Inc., 1997. http://www.openapplications.org/

37Osório, A.L., Oliveira, N.and Camarinha-Matos, L.M. - Concurrent Engineering in Virtual Enterprises: The extended CIM-FACE architecture, *Proc. Of BASYS'98 - 3rd IEEE/IFIP Int. Conf. On Balanced Automation Systems, Intelligent Systems for Manufacturing* (Kluwer Academic), pp. 171-184, ISBN 0-412-84670-5, Prague, Czech Republic, Aug 98.

Papadopoulos, G. A. and Arbab, F. – Coordination models and languages, in Advances in Computers, Academic Press, August 1998, Vol. 46: The Engineering of Large Systems.

Pinto, M. P. T. – Os sistemas de gestão de fluxos de trabalho na coordenação de processos da empresa estendida de construção civil, Tese de Mestrado, FEUP, Oct. 1999.

Putnik, G. D.and Pithon, A. J. – An activity-based model of concurrent engineering system, in Proceedings of BASYS'02 – Knowledge and Technology Integration in production and services, Kluwer Academic Publishers, ISBN1-4020-7211-2, 2002.

Rabelo, R. and Camarinha-Matos, L.M. - Negotiation in Multi-Agent based dynamic scheduling, Int. Journal on Robotics and CIM (Pergamon), Vol. 11, N. 4, Dec 1994, pp. 303-309.

Rabelo, R., Camarinha-Matos, L.M.and Afsarmanesh, H. – Multi-agent perspectives to agile scheduling, Proc. Of BASYS'98, Intelligent Systems for Manufacturing (Kluwer Academic), Prague, Czech Republic, Aug 98.

Rabelo, R., Camarinha-Matos, L.M.and Afsarmanesh, H. - Multi-agent-based agile scheduling, Journal of Robotics and Autonomous Systems (Elsevier), Vol. 27, N. 1-2, ISSN 0921-8890, Apr 1999.

Rabelo, R., Camarinha-Matos and L.M.; Vallejos, R. - Agent-based brokerage for virtual enterprise creation in the moulds industry, in E-business and Virtual Enterprises, Kluwer Academic Publishers, Oct 2000.

Reichert, M., Hensinger, C.and Dadam, P.: Supporting Adaptive Workflows in Advanced Application Environments. Proc. EDBT-Workshop on Workflow Management Systems, Valencia, Spain, March 1998, S. pp. 100-109.

Rocha, A.and Oliveira, E. – An electronic market architecture for the formation of virtual enterprises, in Camarinha-Matos, L.M.and Afsarmanesh, H. – Infrastructures for Virtual Enterprises - Networking Industrial Enterprises, Kluwer Academic Publishers, 1999.

Schlenoff, C., Gruninger M., Tissot, F., Valois, J., Lubell, J. and Lee, J., 2000 - The Process Specification Language (PSL): Overview and Version 1.0 Specification, NISTIR 6459, National Institute of Standards and Technology, Gaithersburg, MD (2000).

Schmidt, K. and Simone, C. – Coordination mechanisms: towards a conceptual foundation of CSCW systems design, Computer Supported Cooperative Work: The Journal of Collaborative Computing, Vol. 5, n^o 2-3, 1996, pp. 155-200.

Shen, W. – Agent.based cooperative manufacturing scheduling: an overview, COVE News N. 2, www.uninova.pt/~cove/newsletter.htm, Mar. 2001.

Shen W. and Norrie, D.H. - An agent-based approach for distributed manufacturing and supply chain management, in Globalization of Manufacturing in the Digital Communications Era of the 21st Century: Innovation, Agility, and the Virtual Enterprise, Jacucci, G. (ed.), Kluwer Academic Publishers, 1998.

Simone, C., Schmidt, K., Carstensen, P. and Divitini, M. – Ariadne: Towards a technology of coordination, Report of Riso National Laboratory, Riso-R-943 (EN), Nov. 1996.

Smith, R.G. – The Contract Net protocol: high-level communication and control in a distributed problem solver. IEEE Trans. on Computer, vol. C-29 pp1104-1113, 1980

Suessmann, B., Colombo, A. W. and Neubert, R. – An agent-based approach towards the design of industrial holonic control systems, in Proceedings of BASYS'02 – Knowledge and Technology Integration in production and services, Kluwer Academic Publishers, ISBN1-4020-7211-2, 2002.

SUN - JINI Technology Architectural Overview, www.sun.com/jini/whitepapers/architecture.html, Jan. 99.

Tølle, M., Bernus, P. and Vesterager, J. – Reference models for Virtual Enterprises, in *Proceedings of PRO-VE 2002 – Collaborative Business Ecosystems and Virtual Enterprises,* L.M. Camarinha-Matos (Ed.), Kluwer Academic Publishers, ISBN 1-4020-7020-9, Sesimbra, Portugal, 1-3 May 2002.

WfMC - Workflow Management Coalition, The Workflow Reference Model - Document Number TC00 - 1003, Issue 1.1, Brussels, Nov 94.

Part V

Case Studies

19

FORD MOTOR COMPANY'S INVESTMENT EFFICIENCY INITIATIVE: A CASE STUDY

James L. Nevins, Robert I. Winner, and Danny L. Reed

Institute For Defence Analyses (US)

19.1 Introduction

This Chapter is based on a document[1] prepared for the Office of the Principal Deputy Under Secretary of Defense (Acquisition and Technology) under the task order Defense Manufacturing Strategy and addresses a task objective, to provide a case study on integrated product/process development implementation. This case study will be used for acquisition and technology training purposes by the sponsor. Many of the incentives, strategies and implementation approaches at Ford have parallels in and implications for the acquisition processes of the Department of Defense (DoD). The DoD student is asked to draw conclusions based on his or her own situation.
Ford Motor Company[2] operates in an intense competitive environment. Its traditional markets are now mature and show only marginal growth. Four in-

[1] This work was conducted under contract DASW01 98 C 0067, Task AD-1-950, for the Office of the Deputy Director, Systems Engineering, Office of the Director, Test, Systems Engineering and Evaluation, Office of the Under Secretary of Defense (Acquisition and Technology). The publication of this IDA document does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agency. (c) 1999 Institute for Defense Analyses, 1801 N. Beauregard Street, Alexandria, Virginia 22311-1772 (703) 845-2000. This material may be reproduced by or for the U.S. Government pursuant to the copyright license under the clause at DFARS 252.227-7013 (NOV 95). Original: Institute of Defense Analyses IDA Paper P-3311 (Revised) Log: H 99-001057 (April 1999) Approved for public release; distribution unlimited. Re-published in this book with permission.

[2] The authors wish to thank the Ford managers who participated in this study: Mr. Gene Nelson, Ford Director, Advanced Manufacturing Pre-Program Engineering, and Ms. Gail Copple, Ford Investment Efficiency Manager. We are grateful to Mr. Barry Lerner, author of related Ford training materials, who clarified many points for us. We also thank Dr. Daniel Whitney, Senior Research Scientist at the Massachusetts Institute of Technology's Center for Technology, Policy, and Industrial Development; and Mr. Russell Shorey, consultant, for their many helpful

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

centives are driving Ford: emergence of new growth markets, smaller-volume niche markets, shareholder returns, and competitive drivers. Growth markets for the auto industry are centered in the Far East, Eastern Europe, and South America. To compete effectively in the next century, automotive companies must leverage their resources to grow their businesses profitably in these markets.

This is why Ford and the major automotive manufacturers place such an importance on Investment Efficiency as a key operating strategy. Investment Efficiency is the ability to simultaneously minimize investment and optimise value for the customer-the goal being to provide the most product for the investment dollar. Ford has developed its own process of Investment Efficiency, centering on improving the compatibility between its product assumptions and its existing manufacturing processes. This process, Product and Process Compatibility, is facilitated by improved communication of product engineering and manufacturing very early in and throughout the product development process. Investment

Efficiency through Product and Process Compatibility addresses the problem of getting development and production costs under control both on individual projects and across projects. Critical aspects of the Product and Process Compatibility approach to Investment Efficiency include organizational and technical implementations:

- Organizational implementation. Upper management has focused on an implementation strategy based on detailed design-production interrelationships and has enforced the use of this strategy. Ford has formed two new groups of experts who have a comprehensive understanding of how manufacturing processes are related to- and affected by designs and of the investment implications of manufacturing processes. These groups work with the design and engineering teams on a car project from its earliest stages.
- Technical implementation. At a technical level, the manufacturing and investment knowledge is captured by design metrics, quantitative targets, and design rules to be used during the integrated engineering process and at milestone reviews.

In discussions with the authors, Ford managers have also provided the following 'lessons learned':

- Changing mind-sets. The Investment Efficiency process required a fundamental change in the mind-set for Ford's product development organization. Because this mind-set is the result of many years of vehicle development, the change in the culture does not occur overnight. Change will be gradual rather than immediate.

---

suggestions and reviews. This document was reviewed by Dr. Richard J. Ivanetich of the Institute for Defense Analyses. This study was conducted during 1996-1997. Ford released the information for public distribution after two years had elapsed.

- Understanding the need for change. Ford's own financial studies showed Ford lagging behind competitors in product development costs. Entering overseas markets required Ford's products to be low cost but still provide outstanding quality and exciting features. At the same time, the mature markets required the introduction of new and innovative products. Thus, the need for Investment Efficiency was clear to both Ford management and its employees.
- Strengthening management support. In establishing the Investment Efficiency Council to oversee the development of process and review progress with the platform teams, Ford signalled its senior management support to its employees. No platform team can go through the development 'gateway', Ford's product milestone review, with an investment status not at-, or near its investment target.
- Creating aligned objectives. Ford is breaking down organizational 'chimneys' through the use of matrix management. Ford is also using the Affordable Business Structure, a planning framework for costs that guides development of vehicles that are both affordable to Ford and its customers. The Affordable Business Structure targets are the common element to align the business objectives of manufacturing divisions and the product development groups. Each organization is charged with getting its cost on target with the Affordable Business Structure.

This Chapter describes how the Ford Motor Company is implementing a management-driven initiative to drive cost tradeoffs and cost targeting very early in its product/ process development process and throughout product realization. The name of this initiative is Investment Efficiency-the ability to simultaneously minimize investment by Ford and to optimise value for the customer. The goal is to provide the most product for the investment dollar. Implementation of the initiative is through the mechanism called Product and Process Compatibility.

As a case study, this Chapter is intended for use by students of the acquisition process in the Department of Defense (DoD). It is intended to elicit thought and discussion of how the Department can better integrate cost tradeoffs and cost targeting into its acquisition processes and integrated process teams.

The contents of the Chapter are based on two visits to Ford during 1995, updates and revisions to the document from Ford management during late 1996, and studies, contacts, and documents going back several years by the authors and others. The authors used this broad base of experience to place the activities in a historical context. Quantitative results are estimates by the Ford managers interviewed for the study or from previously published studies. Although the reported data were not independently verified, these stories have been subjected to the scrutiny and judgment of the authors and reviewers.

Organization of this Chapter is as follows:

- Section **19.2**, 'The Need for Investment Efficiency at Ford', discusses the four incentives that drove Ford to Investment Efficiency: emergence of new

growth markets, smaller-volume niche markets, shareholder returns, and competitive drivers.

- Section **19.3**, 'The Investment Efficiency Process', gives a brief background on Investment Efficiency at Ford before the Ford 2000 reorganization began. It then defines and describes what Investment Efficiency is today at Ford.
- Section **19.4**, 'Basic Targets of Investment Efficiency at Ford', compares previous and current methods of costing products.
- Section **19.5**, 'Strategies of Investment Efficiency', describes the strategies that led up to and now include Product and Process Compatibility, a strategy of arriving at the best product and process concept by simultaneously optimising four main drivers of investment: Reusability, Commonality, Carryover Product, and Complexity Reduction.
- Section **19.6**, 'Product and Process Compatibility Tools', discusses Investment Efficiency Metrics, Manufacturing Design Rules, Generic Product/Process Concepts, and Life Cycle Cost Analysis-all used to drive Product and Process Compatibility.
- Section **19.7**, 'Future Small Car Program Pilot', contains a description of a pilot program using Ford's Investment Efficiency initiative.
- Section **19.8**, 'Organizational Changes at Ford', describes the changes in management and in Ford's relationship with its suppliers.
- Section **19.9**, 'Lessons Learned', is a compilation of lessons reported by Ford managers.

These lessons are grouped into four areas: changing mind-sets, understanding the need for change, strengthening management support, and creating aligned objectives. Section 10, 'Discussion Items', lists issues and questions pertinent to students of the DoD acquisition process.
At the end of the Chapter a bibliography, a glossary or specific terms used in this chapter, and an acronym list are given. The Chapter also includes a list of discussion ideas to further explore the concepts presented. These discussion ideas can be used with students in a classroom or interactive setting.

19.2 The Need for Investment Efficiency at Ford

Ford is a very large, U.S.-based international manufacturer of cars and trucks. It is one of a few dominant companies in its industry. Until the early 1980s, Ford and its domestic competitors had little serious competition from abroad. But the expansion of international companies into U.S. markets, in combination with other external events, created radical shifts in consumer options and preferences. In the United States, government became more involved in consumer protection and industry regulation. These shifts shocked the domestic automobile industry from a position of market leadership to one in which growth, sales, and profitability were severely challenged.

During the period of the mid-1980s through late 1996, Ford has increasingly broadened the scope of its initiatives to address these challenges. First there was a concentration on improving quality. Then Ford mounted the effort to integrate product engineering and manufacturing process design during the engineering phase. Also during the period, Ford adopted and developed ways of improving its approach to defining product requirements from the customer's viewpoint, and it began a revolutionary shift in how it dealt with suppliers as team members rather than adversaries. During the 1990s, Ford began to implement very broad, systematic approaches to cost savings in the Product Development process and to institutionalise the process.
During the early part of this decade, it became apparent to top Ford management that its product development costs were not competitive with the best in the industry. In addition, the automotive industry landscape was changing rapidly, driven by the emergence of new markets. Four incentives drove Ford to Investment Efficiency: emergence of new growth markets, smaller-volume niche markets, shareholder returns, and competitive drivers. Each incentive is discussed in further detail in the following subsections.

19.2.1 New Growth Markets

Beginning in the 1980s, the world's major automotive manufacturers realized that the growth rate in its traditional markets (North America, Europe) had slowed rapidly. These markets were saturated with overcapacity, and future growth in these markets was determined to be minimal.
However, the late 1980s saw political upheaval around the globe, with new democracies being born around the globe. Suddenly, new markets for the automotive industry were appearing on the landscape: the Far East, South America, and Eastern Europe. Ford and the other major automotive manufacturers realized that these markets were the areas of future growth. Now the challenge to Ford-and to most automotive manufacturers-is how to develop low-cost vehicles based on low product investment. Cost is a major barrier for consumers in those markets.

19.2.2 Smaller-Volume Niche Markets

The small amount of growth in the mature markets interacts with the customers' interest in products that result only in the creation of niche segments. Today's automotive customer is demanding more niche products. Many of these niche markets are for vehicles produced in relatively small volumes. Ford had to develop a process that would allow it to produce vehicle products in small numbers with efficient investment levels to realize a profit.

19.2.3 Shareholders Returns

The ultimate incentive for Ford as a corporation is to produce an attractive return to its shareholders. Ford's ability to generate profit from its automotive

business is the basis by which it can reward its shareholders with increased dividends.

19.2.4 Competitive Drivers

All the world's major automotive manufacturers are seeking to reduce their cost base and to improve value to the consumer. In the 1980s, quality served to be the competitive advantage for automotive manufacturers. Those manufacturers with high quality products were able to gain market share by providing consumers with reliable vehicles. By the 1990s, the quality gap between the major automotive manufacturers had narrowed, and customers expected high quality as a 'given' when they purchased a vehicle.
Now the major competitive advantage of the 1990s is cost and value to the customer. Companies that can give the consumer the most product for the transaction price provide the ultimate value. The major automotive manufacturers are streamlining their organizations and product development processes to squeeze costs out of their systems. They have learned that costs can be streamlined out of their product development systems by reducing product-related investment. This is being accomplished by reducing vehicle complexity and uniqueness, sharing components across models, carrying over commodity- type parts that provide little or no differentiation to the consumer, and designing new products that reuse manufacturing equipment and processes. Specific examples:
Toyota has led the effort in investment efficiency actions:

- The new Corona and Carina models share parts.
- RAV4 shares 40% of its key parts with other Toyota products.
- Model variations have been reduced by 30%.
- The number of key parts has been reduced by 40%.
- New models target 70% in carryover parts.

Nissan has also set aggressive targets for investment efficiency:

- A committee has been established for model and parts reduction.
- The number of chassis types was reduced from twenty to fourteen, with an estimated savings of $1 billion.
- The number of platforms has been targeted to be reduced from thirteen to six or seven by the year 2000, with an estimated savings of $2 billion.
- The number of parts has been reduced by 50%.
- The number of model variations has been reduced by 30%. Parts commonality is now 60%.
- The Laurel sedan has 20% fewer parts than the previous model and it shares 50% of its parts.
- The Largo minivan shares 45% of its parts.

General Motors is also aggressively employing investment efficiency actions in its product development process. Specific reductions include:

- Total number of car platforms from twelve to five
- 9 engine families to 5
- 10 air conditioning types to 6
- 6 steering column groups to one
- 32 seat types to 4
- 24 starters to 10
- 12 batteries to 5
- 2,700 electrical connectors to 750
- One-third reduction in panels per car

The General Motors 1993 Annual Report states that "cost savings and efficiencies [...] are enormous". It predicts "further economies of scale [...] as we decrease [...] vehicle architectures and increase production volume".
The published results of the streamlining efforts of these three automotive manufacturers alone indicate the broadness of these activities. They range from mandating a common assembly architecture for cowl and dash joint structures to reductions in model variations and in the number of key parts. Achieving the objectives cited will neither affect product performance nor reduce customer expectations. Rather, these objectives emphasize the historical imperative of industrial management who had tried for decades to focus design activities on product design features that were unique as far as the customer is concerned while minimizing the resources expended on the non-unique features. For example, industrial management experts have questioned having nine engine families, forty different electronic key fobs, or twenty-one different radiator caps where a few might do just as well. It could also be argued that all these items are mature technologies so one is no better than another. Why then have nine engine families when five will provide the performance differentiation needed in the market place?

19.3 The Investment Efficiency Process

19.3.1 Introduction

Cars are complex. They consist of several subsystems, each requiring substantial development and manufacturing investment. In 1980, Ford took up to sixty months to develop a new model. Powertrain programs (Engines/Transmissions) took up to seventy two months. Studies have shown that the average development in the U.S. car industry in the 1970s and early 1980s took substantially longer than that of the Japanese. During this time, the resulting U.S. products were viewed as deficient to the imports in overall product quality.
The manufacturing and final assembly facility for a product in the automotive industry takes up more than 600,000 to one million square feet, not counting facilities in the supplier chain. Each semi-customisable product is produced

in quantities that exceed that of typical defense systems by a great deal, but many of the management lessons learned by Ford are applicable in the defense context Ford buys materials, assemblies, and subsystems from internal divisions and a large supplier network. Suppliers range in size from small machine shops to full-service suppliers, responsible for designing, engineering, and production of vehicle systems. As full-service supplier relationships were developed, Ford found it necessary to have computer-based information design systems for both business and engineering purposes.

Generating a product as complex as an automobile is a time-consuming task. There are milestones or 'gateways' where program reviews are held and product/process plans are measured on the basis of product features versus costs to achieve. Over the years, the auto companies have tried to achieve Investment Efficiency with arbitrarily set cost targets and no real control mechanisms. Ford has found at critical review points (about forty-one months prior to large-scale production) that typical programs' projected costs were two to three times more than the costs that were affordable to Ford and the customer; this phenomenon occurred irrespective of program size. The program teams would then spend the next several months scrambling to change product assumptions and features to drive down to their investment targets. This process of 'thrifting' the product (removing features and options) wasted engineering resources, risked product development timing, and had an adverse effect on quality. In addition, this process did not guarantee efficiency because it only focused on driving an in-process design down to a cost target.

19.3.2 The Ford Product Development System (FPDS)

In January 1995, the 'Ford 2000' reorganization was launched to improve all of Ford's practices. The Ford Product Development System (FPDS) was created as part of Ford 2000 and established to re-engineer the Product Development System. FPDS is a cross-functional process that involves all Ford activities and suppliers. As with similar initiatives in other companies, it seeks to improve quality, cost, and time to market. In addition to engineering and cost improvement, Ford has focused on other enablers to achieve significant improvements in quality and cost under FPDS. One of the more significant enabling tools is the incorporation of new computer-assisted design/ manufacturing/engineering (CAD/CAM/CAE) processes within Ford, which are linked to its suppliers.

19.3.3 Investment Efficiency Goals

Investment Efficiency is a subprocess of FPDS and is the ability to minimize investment and optimise value for the customer simultaneously. The goal of Investment Efficiency is to provide the most valuable product for the investment dollar. Figure 19.1 is the way Ford depicts the projection of total project cost at various times during the life of a development project. The top curve

is what Ford has typically experienced in the past. The bottom curve is for the process resulting from current initiatives. The goal of Ford's Investment Efficiency process is to control the development of product assumptions earlier in the development process. As shown in Fig. 19.1, at the critical review points prior to approval of the project, the team's status must be within 20% of its affordable cost target.

Figure 19.2 is Ford's depiction of the drivers necessary for the new process to converge to affordability faster and better than the old process.

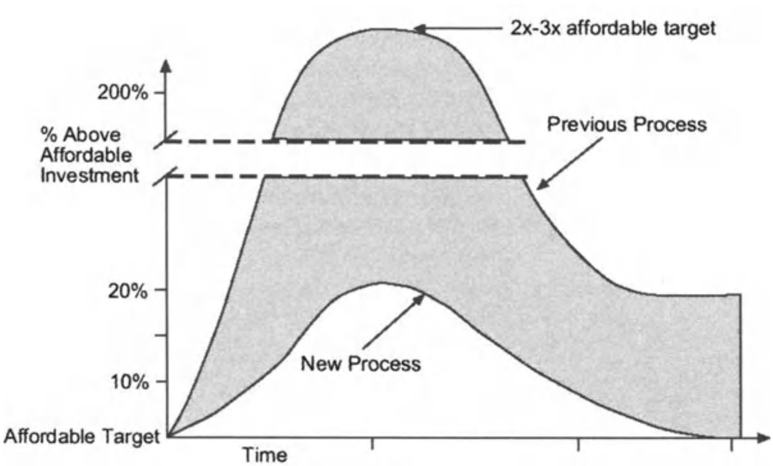


Fig. 19.1. Project Cost Projections Over Time

Ford's experience is that designers seem to have limited knowledge of the physical effects to the manufacturing sites that result from their designs. Thus at the product concept level, the problem currently is *direction*, not complete cost accuracy. Ford is trying to avoid designs that are totally incompatible with their manufacturing facilities. The focus of the Investment Efficiency process is to avoid changing parts that the customer does not see and that do not have quality deficiencies. When Ford does change a part, an objective is to reuse the manufacturing equipment and use that part across several vehicle models (for example, Taurus, Explorer, Continental). Money and resources freed up by this approach can then be focused on growing Ford's business in new markets and paying attractive dividends to shareholders.

The design process enabled by Investment Efficiency has an overriding approach to metrics with a goal of producing the best-in-class products at an affordable cost. Ingredients in this approach include the following:

- Defining targets and metrics up front, based on the Affordable Business Structure (discussed further in Section 4) and engineering metrics (for

example, carryover parts and processes).

- Tracking metrics and targets versus assumptions through the process.
- Understanding the manufacturing drivers for investment.

After more than a decade of shrinking to hike productivity and efficiency, U.S. companies in general are now eager to wring more profits out of these streamlined operations. Ford personnel indicated that they have achieved a level playing field. But the competition goes on.
Note: Achieving a 'level playing field' may be exactly the thing that would be the best result for the Department of Defense and its contractors. That is, achieving efficiency for routine capabilities (and thus saving money and resources for distinctive performance gains) is a reasonable objective.
The most recent Ford initiatives reported here could be viewed as a broad implementation of what the Department of Defense has termed CAIV (Cost As an Independent Variable, in US DoD (2002)) but focused on more than just unit costs. These new initiatives do not replace other initiatives (like concurrent engineering and Total Quality Management) but become elements of Ford's larger strategy built on integrated product/process development (IPPD).

Fig. 19.2. Achieving Cost Targets

Ford's more recent experience is that the product realization process, for typical evolutionary product cycles, has been speeded up by nearly 50% (about thirty to forty months). The pilot for a future small car program has resulted in savings of up to 50% (on the order of $600 million) of the product/process

development cost for a new small car program. At the same time, it has provided maximum value to the customer and minimum investment cost to Ford. (Details of this pilot program are contained in Section 19.7.) It should also be noted that Investment Efficiency is not the only FPDS improvement underway. The entire effort has resulted in reducing the product design cycle and cutting in half the size of a typical vehicle program team.

19.4 Basic Targets of Investment Efficiency at Ford

In this Section we discuss how Ford relates price, cost, investment, and profit. We introduce Ford's categories of investment costs and the resulting investment efficiency targets.

19.4.1 Derivation of Cost Targets

In the past, Ford's Product Development process for a new vehicle program consisted of developing product designs; estimating the tooling, facilities, launch, and engineering costs for those designs; adding a profit margin; and thereby determining the revenue target for the product. This relationship is shown in the following 'classic' product development equation:

$$ProductCost + ProfitTarget = RevenueTarget(PricetoCustomer) \quad (19.1)$$

This process was not substantially different from how other automotive manufacturers developed their products. The focus was on the company and not the customer. Automotive manufacturers, in general, believed that the customer would accept a higher price on the basis that the product had more features and functional improvements as compared to the previous model it replaced.

This practice sent vehicle prices on a steady climb year over year. Whereas most technically advanced products (e.g., calculators, computers) have fallen in price over the years, the costs of automobiles have steadily increased. The average price for an automobile has risen over the past fifteen years from \$10,000 to \$20,000. Some of this increase has been attributable to the costs of adding regulatory-based equipment to the products (e.g., emissions related, air bags, strong bumpers, door panel stiffeners) and economic increases.

As prices continued to increase through the 1980s and early 1990s, customers began to find that they could not afford the payments associated with the purchase of a new automobile. Increasingly, the customers turned to the used car market, and automotive manufacturers were forced to offer costly rebates and lease incentives to attract consumers to their products.

The automotive manufacturers, including Ford, came to the realization early in this decade that they could no longer pass their costs on to the consumer. The concept of value permeated the industry, and the automotive manufacturers needed to address this message from their consumers. Ford's response

was the Affordable Business Structure. The Affordable Business Structure is a planning framework for costs that guides development of vehicles that are both affordable to Ford and its customers. The focus of Ford's cost structure has been fundamentally changed from an internal focus to one where all costs revolve around the price the consumer is willing to pay for a product. The Affordable Business Structure serves to align and link all functional objectives around this concept: the consumer is king and Ford must direct all its effort to producing products that provide the most customer- perceived value for the dollar.

The Affordable Business Structure allows Ford to have funds available to invest in new product programs, weather economic downturns, grow business profitability in new markets, and pay shareholder dividends. It has specific metrics for financial measurables (return on sales, assets), while maintaining a focus on producing world-class products with world-class quality.

The Affordable Business Structure equation differs from the 'classic' approach to product development by shifting the basis of the product development equation from the company to the consumer. As shown below, the basis for product development is the amount the consumer is willing to pay (market price). From the market price, Ford deducts its profit target for the product. The fallout of this equation is the affordable cost.

Simply stated, the consumer dictates the product cost for Ford.

$$MarketPrice - ProfitTarget = AffordableCost \tag{19.2}$$

The Affordable Business Structure equation therefore drives each cost element on the vehicle program's income statement. Both variable and fixed costs are outputs of this equation.

19.4.2 Investment Elements

Ford delineates investment costs for a product program into four categories: Tooling, Facilities, Launch, and Engineering costs[3].

- The Tooling category covers costs for equipment in Ford or vendor plants that generally is permanently modified to cater to a particular Ford product, is specifically designed and life limited to the part it produces, touches the part being produced, or is readily relocated. Ford pays lump sum funds to its internal divisions and outside vendors for such equipment. Examples of tooling include stamping dies, welding fixtures, and molds.
- The Facilities category covers costs for ...

[3] Ford uses the term 'affordable' as a function of product development and manufacturing processes as well as a function of market demand. The reader should compare this with the notion of affordability in the defense world as expressed, for example, in (Gansler , 1993).

- The Launch category covers costs for all related expenses incurred during the launch of a new vehicle program. Examples would include such things as 'offstandard labour hours', training costs, and build-ahead-of-prior-model costs.
- The Engineering category covers costs for vehicle development related to design development, product-related testing, and prototypes for engineering testing.

19.4.3 Key Areas of Investment

Based on historical figures for a new product program, Ford determined that the highest percentage of investment costs were related to Facilities and Tooling (F&T), then Engineering, and finally Launch. Based on this analysis, it was apparent that Ford needed to attack its Tooling and Facilities costs. Historical figures also indicated that within the Tooling and Facilities costs for a new product program, two-thirds of the costs were related to the Body Structure/ Stampings area and the Powertrain Systems (Engine and Transmission related). These were determined to be the high leverage targets for Investment Efficiency.

19.5 Strategies of Investment Efficiency

This Section describes how Ford arrived at its Investment Efficiency Initiative based on results of past efforts. The implementation approach-Product and Process Compatibility-is described along with its four main drivers and their targets. Ford's previous initiatives lacked the following:

- They did not include the creation of an upper management strategy, the incorporation of Investment Efficiency Metrics, nor the discipline to enforce the use of the new metrics at the 'gateways' (i.e., milestone reviews).
- They were not based on special Design Rules and Investment Efficiency Metrics in a form easily understood by the product design personnel who are not familiar with the details.
- They lacked special groups, familiar with the richness of the details, to help the product designers apply these required new metrics[4].

Investment Efficiency is based on three main strategies, listed in the order in which they were implemented at Ford:

- Micro-engineering: Improving cost characteristics of completed component designs (early to mid-1980s.)

[4] Ford's Director of Advanced Manufacturing Pre-Program Engineering hopes that sometime in the future product designers may not need this assistance because they will be more broadly based and thus sensitive to costs and manufacturing issues. Today the system would fail without the help of special groups of experts.

- Simultaneous engineering: Manufacturing, assembly, and design engineers working together during the design of components (late 1980s.)
- Product and Process Compatibility: Making sure product designs are compatible with existing manufacturing processes, tooling, designs, and facilities early in the design process (as of this writing.)

As is common in U.S. industry, the earliest implemented strategy is applied the latest in the development process. Product and Process Compatibility is applied very early in the design process but is being implemented last. Each strategy is discussed in further detail in the following subsections.
Investment Efficiency has three implementation features of note:

- The formation of two new groups, Advanced Manufacturing Pre-Program Engineering (AMPPE) and Investment Efficiency and Competitive Analysis (IE&CA). AMPPE is composed of senior manufacturing engineers and is the voice of manufacturing very early in the design process. IE&CA is composed of engineers and finance people and is responsible for the design of the Investment Efficiency process. It also aids AMPPE in the development and implementation of metrics.
- Design rules and metrics for achieving Investment Efficiency for the major auto systems. These rules and metrics provide the detail criteria for guiding the product design process.
- Management requirements that these Investment Efficiency metrics be used at the major Product Design Review 'gateways' (milestones).

Ford has been using both Micro-engineering and Simultaneous Engineering since the late 1980s. It has found that these processes could yield about 10 to 20% savings in product development costs. However, these savings were not enough for Ford to meet its investment targets and provide optimal value to its customer. These processes occurred too late in the development process to drive major reductions in the development costs for the product.
Ford needed a strategy that would bring together product and manufacturing engineers earlier in the product development process in order to understand and control the drivers of investment for the product. Product and Process Compatibility is the strategy that Ford has developed to make this happen. Figure 19.3 is Ford's depiction of the relative leverage that the three strategies have on costs.

19.5.1 Micro-Engineering

Micro-engineering is a design-process strategy within Ford whereby product teams look at a completed part design to identify tooling opportunities either at the Ford assembly plant or the vendor manufacturing site. The process focuses on optimising a completed component design to identify tooling opportunities either at the component or assembly process. Micro-engineering occurs latest in the development process. It initially involved the elimination

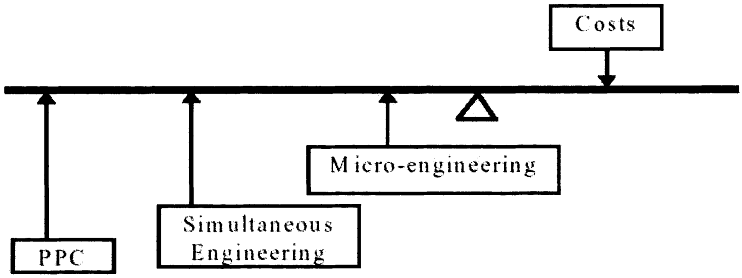


Fig. 19.3. Strategic Leverage

of all Ford-unique requirements on purchased facilities and tooling. Instead, the Investment Efficiency initiative substituted American Society for Testing and Materials standards for Ford-specific requirements where possible. An example of Micro-engineering would be a detailed review of standards and specifications for a component to eliminate unnecessary costs to the customer.

19.5.2 Simultaneous Engineering

Simultaneous Engineering[5] is a design-process strategy within Ford whereby both Manufacturing and Product Design engineers work together during the design of a component. The role of the Manufacturing engineer is to provide ideas on making designs 'friendly' to manufacture to the Product Design engineer. The role of the Product Design engineer is to take into account system effects, product attributes, and warranty/customer input during the design phase.

An example of Simultaneous Engineering would be modifying the design of a component to enable easier ergonomics for installation.

19.5.3 Product and Process Compatibility

Micro-engineering and Simultaneous Engineering remain key elements of Ford's overall investment efficiency strategy. Both strategies serve as the foundation for Product and Process Compatibility. Following the two earlier strategies, the next natural step was to bring product design and manufacturing engineering together at the initiation of the product development process.

Product and Process Compatibility is arriving at the best product and process concept by simultaneously optimising four main drivers of investment:

[5] Ford was an early adopter of simultaneous engineering practice and influenced early DoD studies on concurrent engineering (e.g., Win89). These studies and others resulted in actions which later developed into DoD's IPPD initiative.

reusability, commonality, carryover product, and complexity reduction. (These drivers are further discussed in the next section.) Product and Process Compatibility is based on both planning and designing products in a manner to control these drivers of investment. The key to investment efficiency is to employ product and process compatibility efforts early in a product development phase before parts are designed and before assumptions are made. The focus of the process is bringing together product and manufacturing engineering early in the development phase to understand the implications of design alternatives on the manufacturing process.

It is easy for a product development process to become focused on customer needs and lose sight of internal manufacturing issues. When designers are required to consider these factors, it will often require more initial engineering skill to find the right balance to achieve the overall goal. Prior to this initiative, and without sustained input on manufacturing issues, it was not unusual to find that manufacturing issues were overlooked. To meet customer requirements today, the product must provide the right combination of features and value for the money.

When the balance of features and manufacturing considerations are overlooked in this initial process, it becomes very difficult to correct the problem downstream. Changes after concept design work often compromise either or both areas. Ford's improved Product and Process Compatibility strategy focuses on providing this balance at the initiation of new program work. While providing a more challenging task to the new program team, this strategy will result in the best product at the lowest cost. This combination assures fewer changes, better timing, and good product quality for the customer.

When applied early in the Product Design process, Product and Process Compatibility has demonstrated a potential to reduce design time by up to 33%, reduce engineering workload, and lower total cost.

19.5.4 Drivers of Investment

This section describes the four main drivers of investment addressed by Product and Process Compatibility: reusability, commonality, carryover product, and complexity reduction. These four drivers interact with each other.

19.5.4.1 Reusability

Reusability focuses on use of existing prior-model tools, facilities, and processes. Reusability minimizes investment by making use of tools and facilities that are already available and that have already been funded by the company.

19.5.4.2 Commonality

Commonality focuses on product assemblies, features, product attributes, and facilities and tools shared with other products. The ability to use a common

part across several vehicle lines enables Ford to avoid spending capital to design unique tooling (thereby reusing the existing tooling), engineer the unique part, and build prototypes of that unique part. Commonality also allows Ford to amortize its tooling cost over a larger number of parts, and it simplifies inventory for the downstream parts and service activities. Toyota has set the benchmark for commonality within the automotive industry. When Toyota decides to change a part, it implements a process to incorporate that design across its model line-up. For example, a recent trip to a Toyota/Ford dealership revealed that Ford had fifteen different unique designs for a certain underbody component across its models while Toyota had one.
Chrysler has also been successful in driving common parts across its product programs. Commonality is one of the keys to Chrysler's drive to become the low cost producer in the automotive industry. Chrysler follows a 'parts bin' approach to developing new model vehicles. For example, the recently introduced Plymouth Prowler roadster was produced for a reported $75 million. This was accomplished by using existing tooled parts from its production vehicles:

- The side view mirror controls, engine, and transmission are from its LH sedan.
- The interior door handles and the gauge faces are out of its Viper sports car.
- The climate controls and rear brakes are from the Neon.
- The plastic vent grilles and front brakes are from its minivans.
- The steering column and turn-signal stalk are from its Jeep Grand Cherokee sports utility vehicle.
- The gear shift lever is from its Eagle Vision sedan.

Ford is aggressively pursuing parts commonality across its vehicle line-up. For example, the recently introduced Ford Expedition shares 50% of its parts with the F-Series truck. Ford's future product plans call for new products to have at least 25% of their parts come 'off the shelf' from existing production vehicles. Ford and most automotive manufacturers have realized that customers are not willing to pay for change just for the sake of change. Therefore, when a high quality component design is established, it is important to drive that design throughout the product line-up.

19.5.4.3 Carryover Product

Carryover product focuses on carrying over product components, assemblies, or features from the prior generation model of a product, particularly those parts that the customer perceives to have little or no differentiating effect on value. This enables reusability of the existing manufacturing equipment.
In the past, it was fashionable within Ford always to design new parts. 'New is better' was the slogan by which engineers consistently churned out new

product component designs. 'Building the better mousetrap' was the goal of Ford's engineering departments. If an existing part were not performing well in the production vehicle, the first instinct was to redesign the part for the new model, rather than fixing the problem.
The culture within Ford's engineering ranks was one of the main drivers of this philosophy. Young engineers would hear glorious tales of how senior Ford engineering managers had risen through the ranks by developing new, higher functioning designs for product components. Over the years, this bred the 'clean sheet' approach to component design that burdened the company with parts proliferation and ultimately led to quality risks as each new product program was launched with a myriad of new component designs.
Ford has drastically changed its view over the last several years. The company has come to realize that customers refuse to pay for change for the sake of change. The focus has to be on only changing parts and vehicle systems from which the customer derives differentiated value. This could include such things as visible items on the interior or exterior of the vehicle or parts which aid in the ride or handling of the product. That leaves literally hundreds of parts on a product that the customer does not see or care about.
Ford Engineering management has also radically changed its views and reward system for its ranks. Senior Engineering management has challenged its employees to create higher functioning parts by simply modifying existing designs rather than creating all-new parts from a clean sheet. Ford recognized that the most difficult engineering accomplishment was to increase functionality from a carryover part, as opposed to starting from a clean-sheet approach.
Ford now realizes the quality benefits of launching products with a higher degree of carryover parts-those that have already been proven in the field to be reliable and defect free and that are easily assembled. Parts will only change now for new products if demanded by the customer or to fix a quality problem with the existing part. Ford's future product programs that are based on 'freshening' of an existing model (for example, the next generation Taurus) have been given the target to utilize a significant fraction of carryover parts in the design of the new model.

19.5.4.4 Complexity Reduction

Complexity reduction focuses on reducing the intricacy of a product or manufacturing process. The ability to reduce part complexity will increase commonality of parts across models.
Ford is moving aggressively to reduce its corporate part complexity. In 1994 it established a Complexity Reduction office that is responsible for studying over one hundred major component commodities and for recommending strategies for reducing the number of unique designs in each grouping. For example, Ford is reducing the number of designs for air extractors from 14 to 5, batteries from 36 to 8, power distribution boxes from 12 to 2, and cigarette lighters from 21 to 1. What this means to Ford's product teams is that if they plan to

change any of these commodities on their program, they must select one of the corporate designs in the 'parts bin' for that commodity. This will eliminate the tendency for each product team to design its own unique part for that commodity.

Reducing part complexity will enable Ford to save money on designing and building unique tooling, reduce costs for engineering testing and prototypes, and increase downstream savings by simplifying the inventory for its Parts and Services division as well as its assembly plants. It will also yield quality benefits as proven existing part designs are used on new products.

Ford has also established overall corporate targets for complexity reduction. Ford has targeted to reduce the number of vehicle platforms (components that make up the underbody of a vehicle) by 50%. It also plans to reduce the total parts in its corporate 'parts bin' by 30% by eliminating unique designs through the aforementioned commodity reviews. All new products to Ford (i.e., new segment vehicles) will be targeted to use parts from the corporate 'parts bin'. 'Freshening' programs for existing products will be targeted to use a larger percentage of parts from the previous model or the corporate 'parts bin'.

Ford also has plans to reduce the manufacturing complexity when it does produce a new and unique component. For example, targets have been established to reduce the number of operations per part for stamped parts, and one of the focuses of Product and Process Compatibility workshops is to design new parts that can be processed through fewer workstations than the previous design. This will allow Ford to reduce its expenditures on its manufacturing equipment, further driving investment efficiency.

19.6 Product and Process Compatibility Tools

Ford's organizational and process restructuring had to address the following issues:

- Understanding that designers of large complex systems do not necessarily understand the cost and investment impact issues involved with detail design. Similarly, their understanding of the constraints and capabilities of manufacturing systems is weak.
- The need for a strategy to control proliferation of product and component variations and to address investment efficiency, market drivers, and available technology.
- Management maturity that realized that detailed cost-process knowledge resident in a corporation (or available for purchase) must be re-packaged and brought forward with experts to the initial product design point.
- Management realization that to implement these policies, new Design Rules and Investment Efficiency Metrics had to be created in order to guide all aspects of product design.

- To impose structure on the development process, these new metrics must be used at the major program review points ('gateways'). Ford has developed four main tools to drive Product and Process Compatibility in its platform teams: Investment Efficiency Metrics, Manufacturing Design Rules, Generic Product/Process Concepts, and Life Cycle Cost Analysis. Each is discussed in the following subsections.

The new strategies, tools, and methods also required new training programs supported by books of metrics, design rules, a training guide, and two videos.

19.6.1 Investment Efficiency Metrics

Product and Process Compatibility is based on the concept of controlling the physical drivers of investment (reusability, commonality, complexity reduction, and carryover product) for a product. A car or truck is an extremely complex product with a myriad of vehicle systems and components. Each vehicle system has its own unique drivers of investment that need to be first identified and then measured through a metric. Ford began the development of its Investment Efficiency Metrics by identifying twenty-six major systems of a vehicle that had unique drivers of investment. Examples include engines, transmissions, seats, instrument panels, brakes, and suspension components. Ford then assembled cross-functional teams of subject matter experts for each system. Team members included product engineers, manufacturing engineers, and vendors. The team's assignment was simple: 'What physically drives investment dollars for your system, and how would you measure these?' Each team identified its physical drivers of investment and categorized them as a driver of reusability, commonality, carryover product, or complexity reduction. Then the team determined a metric for tracking that driver.
Each team reviewed its findings with an oversight committee, the Investment Efficiency Council, consisting of senior management from Product Development, Manufacturing, and Purchasing. The information was then finalized and made available for product teams.

19.6.1.1 Ford's Metric Process

Investment Efficiency Metrics are the cornerstone of Product and Process Compatibility efforts at Ford. They are the tool by which product teams control the drivers of investment for their product program. Table 19.1 summarizes the process. The process begins shortly after team formation during the Pre-Strategic Intent phase.

Table 19.1. Metrics Process

Step	Action	Objectives
Pre-Strategic Intent:Review Manufacturing Knowledge Base[6]	Identify limitations of existing manufacturing equipment and product.	1. Identify vehicle areas expensive to change. 2. Avoid manufacturing changes. 3. Make informed decisions that affect manufacturing costs.
Set macro targets.	Set high-level targets for major physical investment drivers.	1. Align early metrics with Affordable Business Structure. 2. Avoid drift to unaffordable assumptions.
Efficiency Council review.	Senior development, manufact. and purchasing managers review macro targets.	Review and consensus.
Set system targets (example systems: body, engine/transmission)[7]	Set physical metric targets for each system.	1. Detail physical metrics. 2. Align total system investment targets in proportion to subsystem investment allocations.
Product/Process Compatibility workshops.	Detail relationship of product designs with physical processes.	Align detailed product assumptions with physical targets per system.
Efficiency Council review.	Senior development, manufacturing, and purchasing managers review detailed targets.	Fix the metrics that must be met throughout development.
Using metrics and targets.	Design teams measure emerging designs using metrics and compare with targets.	Meet the targets in the actual design.
Review designs.	Subteams report status versus targets to project manager at regular team reviews.	Keep the team on target.

This is the highest leverage point for Investment Efficiency for a product as the assumptions are just being developed. The product team reviews the Manufacturing Knowledge Base, a compilation of information regarding the 'hardpoints' of the existing product and the assembly equipment put together

[6] the Manufacturing Knowledge Base was created and is maintained by the Advanced Manufacturing Engineering Group.

[7] 'Systems' here are equivalent to 'subsystems' in DoD parlance

by Ford's Advanced Manufacturing Engineering group [8]. This data enables the team to identify the areas of the vehicle that will be expensive to change because of limitations of the existing manufacturing equipment and the product.

This upfront knowledge allows the team to tailor its product assumptions to make changes to meet customer needs for the product without major changes to the existing manufacturing process. Under this approach, if a team does decide to make a change that will drive significant costs, it understands the effect of that decision.

Targets for these physical drivers of investment are based on the investment dollars allocated to the program from the Affordable Business Structure. For example, a $500 million product program will have a much higher percentage of Assembly Line Equipment Reusability (e.g., 90%) than a $1.5 billion product program (e.g., 60%). The goal of these early metrics is to get alignment of the macro product assumptions with the Affordable Business Structure dollars early in the development stage. This will ensure the product team does not 'drift' into creating product assumptions that are deemed unaffordable later in the development process. This is the most crucial point in Ford's Investment Efficiency process: It fundamentally sets the product program's macro assumptions in line with what is affordable to the company-and ultimately the customer.

Note: The auto industry has a long history of striving for standard processes and designs, benchmarking (both internal and external), and formalizing this information into weighty (10-inch thick) books. For example, there are such books on standard design and processes for designing and manufacturing engines, stampings, transmissions, and gear tooth stresses.

The team begins the process by setting macro targets for the major physical investment drivers for the product. Examples of metric targets set during this stage include the following:

- Reusability of assembly line equipment
- Floor space utilization in assembly plant
- Reusability of body construction and powertrain tooling and facilities
- Percentage of carryover parts utilized
- Percentage of common parts (from Ford's parts bin) utilised

The problem is not a lack of technology but rather not enough process management. That is, methods are required for standardizing processes like common manufacturing locators and weld-lines within and between auto programs (refer Table 19.2 for examples of how the metrics develop through the product development phases).

[8] 'Hardpoints' are specific locations on the vehicle's body pan to be used as attachment and reference points for all tooling, welding, and assembly for the entire vehicle. The approach being adopted across the vehicle industry is that there should be four hardpoints in common for all products in a company's line.

[9] Note: The phases are not precisely thus named at Ford.

Table 19.2. Investment Efficiency Metrics By Phase[9]

General Targets	**General Targets**	**Specific Targets**	**Specific Targets**	**Objectives**
% Part Reduction Carryover Parts Common Parts	%Part Reduction Carryover Parts Common Parts	%Part Reduction Carryover Parts Common Parts	Count(Abs.) New Parts Carryover Parts Common Parts	Count(Abs.) New Parts Carryover Parts Common Parts
Number of Platforms	Specific Platform & Range Hardpoints	Specific Platform & Range Hardpoints	Specific Platform & Range Hardpoints	Specific Platform & Range Hardpoints
Powertrain Combinations	Powertrain Combinations	Powertrain Combinations	Powertrain Combinations	Powertrain Combinations
	Labour Hours (% Less Than Carryover)	Labour Hours (Absolute)	Labour Hours (Absolute)	Labour Hours (Absolute)
		Initial Variable Cost & Investment	Variable Cost & Investment (Absolute Targets by Operation)	Variable Cost & Investment (Absolutes by Operation)
	Body-in-White Combinations	Body-in-White Combinations	Body-in-White Combinations	Body-in-White Combinations
			Stamping Operations Per Part	Stamping Operations Per Part
			# of Close-out Welds	# of Close-out Welds
			Reusability of Processes	Reusability of Processes

After developing its macro physical targets, the team presents them and the workplans to the Investment Efficiency Council for review and consensus. After the Strategic Intent Phase, the product team begins to form subteams, each of which is responsible for a vehicle system on the product (i.e., Engine/Transmission Team, Interior Team, Body Structures Team). These teams go through a metric target setting process for their vehicle systems. For example, the Body Structures team would be responsible for having physical targets established for its vehicle system. The teams set their physical targets at a level that is comparable to the amount of total vehicle investment

dollars that has been allocated to them. An example of metrics for the Body Structure team is shown in Table 19.3.

Table 19.3. Body Structures/Stampings Investment Metrics

Driver	Metric
Total # parts/vehicle	# parts
Annual volume	Volume/capacity—A, B, C Dies
Commonality/carryover	% carryover parts % common parts % carryover process
Dies/part # operations per die set	# exceeding four operations # dies/part # double attached % progressive dies
Common architectures % common joints	% carryover locators % re-use of tooling
Common assembly	# load-weld-load % common assembly sequence % common tools for derivatives
Reusable tooling	# carryover stations # carryover re-spot # carryover inspection stations
Integrated build	Additional floor space list units
Labour/unit	Hours/vehicle

Note: These metrics are at a detail level with respect to the manufacturing process. This example illustrates the flow-down from the high-level investment efficiency strategy to the detailed metrics applied to product/process designs. It is important to understand that early cost estimates are necessarily imprecise. Ford's management appreciates that product concept targets and assumptions may not be rigorous, but as the program matures, metrics should continuously drive towards affordable targets so that by program approval clear verifiable metrics exist, as Table 19.2 illustrated previously.

The product teams begin a series of Product and Process Compatibility workshops to drive the designs of vehicle subsystems to the physical targets established. The objective of these sessions is to align the detailed product assumptions with the physical targets established for each system.

The team then returns to the Investment Efficiency Council to review these more detailed targets and the results of their Product and Process Compatibility workshops. The metrics are used throughout the product development process. As designs are established for vehicle systems, the effects to the phys-

icals are measured against the targets for each system. Each subteam reports its status as compared with targets on the physicals to the project manager during regular team reviews.

19.6.1.2 Milestone Review Process

The milestone schedule has been compressed, but exit criteria remain essentially the same. What has fundamentally changed is the management of the review activities. Major milestone reviews are now carried out by a new Oversight Committee composed of the Vehicle Center's Vice Presidents. Previously, designs were reviewed up to Program Definition by Design Managers not totally familiar with costs or investment issues. Besides changing the management of the review process, management has now dictated that Investment Efficiency Metrics be used at all major program reviews. The new process requires that Investment Efficiency Metrics and product design rules be used at the front end of the product development cycle.

19.6.2 Manufacturing Design Rules

Ford's Manufacturing Design Rules identify the critical parameters or constraints that make up the manufacturing processes and that are the drivers of Facilities and Tooling investment. The rules provide the basic knowledge to assist product designers and engineers to drive towards the Investment Efficiency Metrics targets that have been established. They serve to educate the designer and product engineer on the manufacturing implications of their decisions.

Ford's Manufacturing Design Rules are grouped by major areas such as Stamping, Body Structures, Powertrain, Electrical, and Interior. The use of the rules will assist Ford in driving towards the following:

- Common assembly architecture
- Common assembly sequence
- Common assembly locators
- Reduction in the number of die operations per part
- Plans for accommodating future models and powertrains

For example, a Stamping Design Rule would indicate to an engineer that it is preferred that holes should be placed on a stamped part on flat surface of the part. It is acceptable to have holes placed on a stamped part surface with less than a 15-degree plane. This will allow a press to come straight down and punch the hole. If the hole is on a surface greater than 15 degrees, a secondary CAM operation will be required to punch and trim the hole. This will incur added manufacturing cost to the product. If a hole has to be on a stamped surface greater than 15 degrees, the rule indicates that the engineer should design an embossment into the sheet metal to allow the press to punch the hole on a flat surface.

As shown in this example, each of Ford's Manufacturing Design Rules indicates to the engineer what is preferred, acceptable, or not preferred. The engineer also is provided with an explanation as to why the costs would increase (e.g., secondary CAM operation required) if the rule is violated.
Examples of Detailed Design Rule Types The following examples provide a brief glimpse of other types of design rules used within Ford's component areas. They are provided merely to indicate the breadth and depth of the initiative. Design Rules for Body Construction:

- Same number of panels and joint/sealer designs, allowing common bodyshop process
- Underbody and structure: best opportunity
- Skin panels: styling flexibility with common joint/sealer designs

Note that 'styling flexibility' is being constrained by an opportunity for cost savings by making joint designs common. Common Assembly Sequence Design Rules:

- Make common across vehicle lines.
- Load small parts in subassembly tools.
- Load large parts in large tools in the initial station.
- Avoid designs requiring sequential assembly, i.e., avoid load-weld-load conditions.

The last bullet means that a sequence in which a number of part loads are followed by a single welding step is preferred to one in which loads are interspersed with welds. Common Locator Holes and Surfaces Design Rules:

- Common for body construction and component assembly
- Common locators:
  - From stamping through assembly
  - Between vehicles on the same platform
  - From present generation to next generation vehicles
- Changes limited to one plane
- Underbody and structures most important

Sample Design Rule worksheets for two of the twenty-six component areas were shown to the study team but not released. They were organized by general Design Rule category (e.g., reusability, commonality).

19.6.3 Generic Product/Process Concepts

Generic Product/Process Concepts provide designs and processes that are 'off the shelf' and that have been optimised for both Product and Manufacturing requirements. These concepts enable Ford to achieve greater commonality across its product line-up and to achieve greater reuse of its manufacturing equipment. Ford has been working in several areas to arrive at common part

designs and manufacturing processes in an effort to reduce costs. Examples include the following:

- Generic body architecture
- Generic body shop
- Instrument panel structures
- Powertrain components (e.g., engine blocks)
- Climate control components

These tie in with Ford's work on creating common vehicle platforms for its product line-up. The concept of common vehicle platforms is the key enabler to achieve Flexible Manufacturing.

Flexible Manufacturing is the ability of manufacturing and product design to respond in the shortest amount of time-to-market changes with products that profitably meet customer needs. The advent of common platforms within Ford will enable the development of distinctly different products derived from a common set of assembly tooling. For example, a Ford plant could produce Common Platform 'A' that includes a sedan, coupe, and mini-sports utility vehicle, all in one plant travelling down a common assembly line. As customer demand shifts from one model to another (say, the sedan to a mini-sports utility vehicle), the mix of product could change quickly to meet market needs. Ford's vision is to use a limited number of core platforms from which multiple derivatives could be launched. This will enable Ford to produce more distinctly styled cars and trucks, for relatively low investment costs, thereby achieving its goal of investment efficiency (the greatest value for the investment dollar). To the customer, the vehicle will appear unique, but the platform components the customer does not see will be common across multiple derivatives. The challenge facing Ford and other major automotive manufacturers is how to implement such a strategy.

Ford defines a platform for a vehicle as three main structural assemblies that make up the underbody of the vehicle: Front End Structure, Front Floorpan, Rear Floorpan. The costs to tool and assemble these complex systems are the most expensive portion of investment for a Ford vehicle program. An internal Ford study in 1995 indicated that it had thirty-two unique platforms within the company. Some of these platforms' dimensions were within millimetres of others. In essence, Ford spent millions of dollars in the past to design unique platforms that were essentially similar in dimension. Beyond the cost for tooling each platform, Ford also expends engineering resources for development and testing.

A 1992-93 Ford analysis showed gains that could be accomplished by a 'Factory of the Future' that incorporated Flexible Manufacturing in a sensible way. Example gains of a possible Factory of the Future were shown for powertrains with manufacturing based on advanced flexible manufacturing cells. An estimate was made that a Factory of the Future would be able to reduce

downstream[10] costs 70 to 80% at each subsequent product changeover for an initial 25 to 35% increase in facilities and tooling over then-current practice. This leads to the issue of bringing downstream cost analysis to bear in the upstream design process, as discussed in the next section.

Ford's drive towards common platform is in the process of being implemented as part of the Ford 2000 reorganization. Ford has successfully aligned its global product cycle plan around the aforementioned platforms. This approach is projected to save billions of dollars for the company over the next several years.

19.6.4 Life Cycle Cost Analysis

Vehicles are cyclical products that must be updated regularly to meet government or corporate regulation and to provide new, attractive features and up-to-date styling. Ford generally categorizes product changes within a cycle as all-new product, major freshening, and minor freshening, based on the degree of change[11].

For example, Ford might introduce an all-new product offering in the 1997 model year with a twelve-year cycle. It will then plan to conduct a minor freshening on that product after four years, a major freshening after eight years, and will replace the product altogether at the end of the twelve-year cycle. Because of the cyclical nature of the product, extensive planning is required in the initial product design to allow manufacturing flexibility for mid-cycle changes at an affordable level. Often this manufacturing flexibility requires additional amounts of capital expenditures at the start of the cycle to realize downstream benefits at the mid-cycle freshenings.

In the past, Ford's Product Development System was heavily focused on optimising costs for the initial product. As costs were pared down, incremental capital requirements for manufacturing flexibility were often the first to be eliminated. Ultimately, this led to struggles at the mid-cycle freshenings: The product teams could not afford to make substantial product changes because of the inflexibility of the manufacturing equipment. The debate was whether to optimise profitability over the near term versus the entire product life cycle. In addition, Ford's performance system was not set up to reward decisions that optimised long-term profitability. The focus for a platform team manager was to optimise profits for the upcoming product change in the cycle. With Ford 2000 and the development of Affordable Business Structure, Ford has shifted its emphasis to product cycle profitability. Ford realizes that flexibility needs to be planned into new product offerings to ensure lower mid-cycle freshening program costs.

---

[10] 'Downstream' here refers to changeover points to later products and not to the downstream costs-logistics, ownership, and maintenance-of the current product.

[11] Note that 'life cycle costs' at Ford refer to the life cycle of a product line such as the Ford Taurus. In DoD, life cycle costs generally refer to the costs of owning, operating, and maintaining systems.

Platform team managers now are responsible for total product cycle profitability. Therefore, their focus has shifted to optimising total product cycle profitability. Senior management has also realised that incremental capital expenditures may be required during the initial program if they can be proven to lower mid-cycle freshening costs (refer Fig. 19.4).

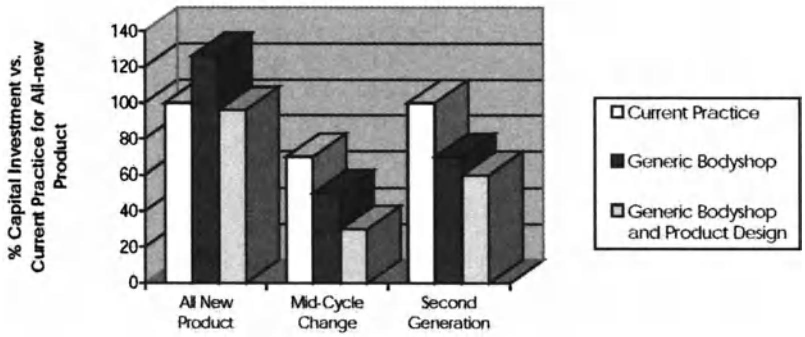


Fig. 19.4. Planned Flexibility Yields Cost Savings During the Life Cycle

As part of the Ford 2000 program, the company has also changed its financial system to place more emphasis on total cost, rather than optimisation of any individual cost element (i.e., Investment, Material, Freight, Labour, etc.). Investment in one element cannot be viewed in isolation, with efforts focused only on reducing that particular cost element. Product decisions will be made that will be in the best interest in maximizing profitability for Ford as a whole.

19.7 Future Small Car Program Pilot

Ford has already piloted its Investment Efficiency process on several future model programs. Pilot results have been positive and the company is now in the process of implementation on all future model program teams. The following is a brief summary of one of Ford's pilot studies, the future small car program.

The future small car program was the initial pilot of Ford's Investment Efficiency initiative. This product will be global in line with the Ford 2000 vision. It will be sold in markets around the world and produced in Ford plants in Europe and North America. The product line-up will include a three-door, four-door, and station wagon. Before Ford 2000 was implemented in January 1995, work had already begun on this pilot program. Initial product assumptions called for unique versions of the vehicle to be produced in Europe and North America. Ford 2000 and its global platform vision challenged the team to create one platform to serve all worldwide markets.

The North American plant's present assembly process was distinctly different from that of the European plants that would be assembling the product. The first approach was to change over the North American plant to the European process to achieve consistency and uniformity in assembling the product. This would result in a new body shop being built at the North American site, alongside the present building, and all new processing equipment to be installed. The present bodyshop at the North American facility had received a major refurbishment in 1989 with new equipment.

This approach was extremely expensive, as an entirely new building would have had to be added to the North American site. In addition, the design of the vehicle had not been optimised for compatibility with the existing manufacturing equipment in the European plants. This resulted in low levels of reusability for the European locations. The product team soon found itself two to three times over its Affordable Business Structure investment target.

The product team dedicated one month to employ a Product/Process Compatibility 'blitz' to lower its investment levels, while still providing the customer with the product requirements being demanded.

The initial phase of the blitz involved identifying the physical drivers of investment for the product. The largest driver of investment was the fundamental difference in the manufacturing process between the North American and European assembly locations. Another large driver of investment was the product complexity among the models, that is, the number of unique parts assumed for each model. The focus of the Product Process Compatibility blitz was to address these two issues.

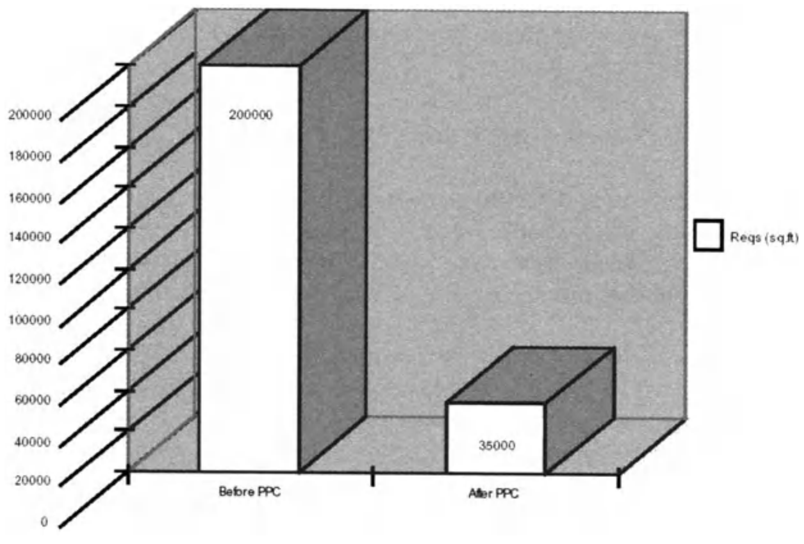


Fig. 19.5. New Building Requirements Reduced 83% (Car Project)

To address the manufacturing process differences, the engineers focused on a new design approach that would enable the product components to be assembled through either manufacturing process. This eliminated the need for the new addition at the North American plant, thereby reducing floor space as well as allowing the product to be processed through much of the existing equipment.

Figure 19.5 illustrates the before-and-after Product/Process Compatibility space requirements. The team also spent time going through each component, to understand how design modifications would allow increased reuse of equipment at all plants. One result of this process was a significant increase of plant processing equipment reusability in Europe and North America.

Figure 19.6 depicts the amount of reuse in construction tooling and facilities.

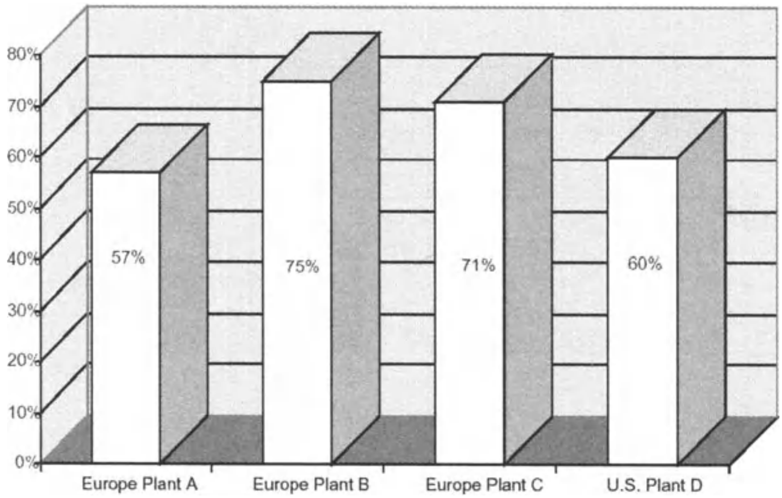


Fig. 19.6. Reuse of Construction Tooling and Facilities (Car Project)

To address the complexity issue among the model line-up, the team constructed a complexity matrix. This matrix listed each component of each model. The team then began a process of examining each component on the model lists to understand why it could not be made common among the model line-up. They found that many of the parts could be made common with minor revisions to the proposed designs. This process drastically reduced the number of unique parts that would have to be engineered and tooled for the product.

Figure 19.7 depicts the results from a similar project to develop a new truck. This truck had three versions with various wheelbases and a different roof height for different markets.

By using Product and Process Compatibility to drive the product's assumptions in line with the existing manufacturing equipment and to reduce product complexity, the team was able to save hundreds of millions of dollars and drive down towards its Affordable Business Structure investment target. This was done without sacrificing product content and features that the customer demands from this product. The team was highly successful in its application of Product and Process Compatibility, and their work has served as the basis for the launch of the process across all of Ford's platform teams.

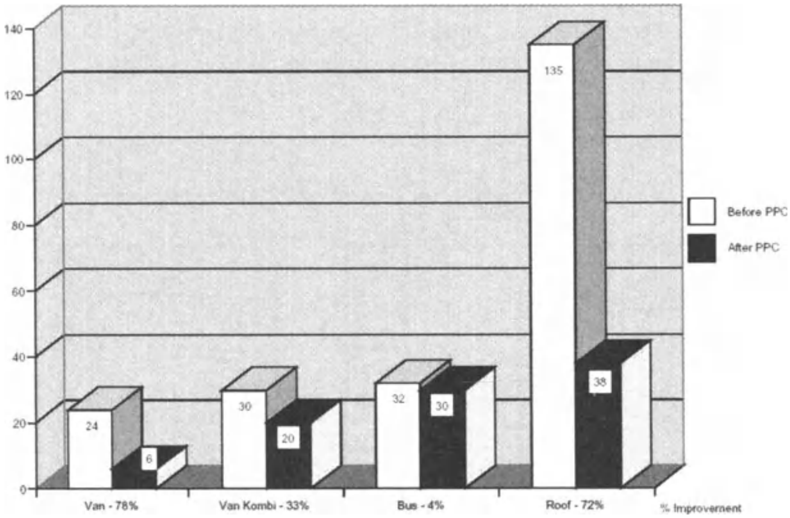


Fig. 19.7. Number of Unique Sheetmetal Parts (Truck Project)

A new and different aspect for Ford in this initiative is the maturing view of management to the larger problems of the Product Development Process and the necessity for management to take an assertive role in defining clear goals and the methods for achieving them, as well as the organizational structure to implement them effectively.
Note: The 'blitz' described in this anecdote is not a 'Tiger Team'. The Tiger Team approach is so situation dependent that very little systematic improvement is gained. Tiger Team results are not transferable to other products because they are too product specific and the processes are too unique (Bowen *et al*, 1994). Here, Ford management has used the small car project as a pilot of a permanent, systemic process change.

19.8 Organizational Changes at Ford

Ford has learned many lessons during the development and implementation of its Investment Efficiency process. Chief among those is that the process will only work with true cross-functional participation. In the case of Ford, this means having representation from such activities as Manufacturing, Product Engineering, outside vendors, Sales and Marketing, and Purchasing.
At the heart of this cross-functional team effort is the relationship between the Manufacturing and Product Development groups. The biggest challenge for Ford was getting these two activities together earlier than ever before in the Product Development phase. In the past, the relationship between these activities was sequential in nature: Product Development developed assumptions and engineered the product and then handed those plans over to Manufacturing for implementation. This relationship precluded effective dialogue between these organizations during the development process, with the result being products that were fundamentally incompatible with the existing manufacturing equipment/processes.

19.8.1 Implementing the Investment Efficiency Process

As part of Ford 2000, two organizations were created to create and implement the Investment Efficiency process: Advanced Manufacturing Pre-Program Engineering (AMPPE) and Investment Efficiency and Competitive Analysis (IE&CA). These groups have representatives on each of the platform teams during the product development process. Their role is to facilitate Product and Process Compatibility efforts on the platform teams.
The AMPPE group is composed of Manufacturing Engineers, all with extensive experience (at least fifteen years). The group is divided into departments based on Ford's assembly process. There are departments associated with Stamping/Auto Assembly, Powertrain (Engine/Transmission) Assembly, and Automotive Components (e.g., Instrument Panel) Assembly. There is also a small department within the group that is involved in Ford's Product Cycle Planning process. The role of AMPPE is to be the voice of Manufacturing on the platform team during the early product development phase (before formal program approval). This includes sharing with the platform engineering team the information regarding existing manufacturing equipment at the assembly site, maximizing reuse of the equipment, and taking advantage of opportunities to reduce manufacturing complexity[12]. The AMPPE representative also works with the platform team to set metric targets to control the physical drivers of investment and facilitates Product and Process Compatibility workshops to drive product design alternatives to the metric targets established.

[12] such as the number of workstations required or the number of direct labour hours required to assemble the product.

The Investment Efficiency and Competitive Analysis (IE&CA) group is composed of both Product Engineers and Finance personnel which have previously served as members of platform teams. Their role is to lead in the creation of Ford's Investment Efficiency process and to aid the AMPPE group in platform implementation (i.e., setting metric targets, facilitating Product and Process Compatibility workshops). Another portion of the group works on tax abatements and incentives with the local municipalities where Ford has established manufacturing sites. The group is also responsible for training Ford personnel on the Investment Efficiency process and for participating in benchmarking studies.

Ford also set up an oversight committee of senior management (the Investment Efficiency Council), to oversee development of the process and conduct platform Investment Efficiency reviews to ensure the process is working. The Council is made up of Senior Management (vice presidents) from Product Development, Manufacturing, and Purchasing, and meets regularly.

In summary, the Ford Investment Efficiency process is based on strengthening the relationship between Product Engineering and Manufacturing early in the product development phase. This is facilitated by the AMPPE and Investment Efficiency groups and overseen by the Investment Efficiency Council. Only when this relationship is strengthened and leveraged will true investment efficiency occur in a product development process.

19.8.2 Involving the Suppliers in Investment Efficiency

As part of a separate activity, over the last several years Ford has completely changed its relationship with Facilities and Tooling suppliers. Instead of fixed-price competitive contracts, Ford has gone to target pricing and negotiated contracts. Target prices are developed from the following:

- Benchmarking: both internal and external[13];
- Looking at similar products (last purchases);
- Historical data;
- 'Business judgment' (For example, are supplier order books filled or empty?)

Suppliers are reviewed, selected (usually one), and invited to join in the design discussions one year earlier then they were in the past (i.e. T-36 instead of T-24 months). They are not paid for this early involvement, except under special circumstances. Since suppliers are involved early, they are also advised of Ford's expectations. In addition, because both parties will be examining the design, Ford expects that the resultant design (at T-24 months) will be the target price minus a 'tad', but that is a negotiated stance. For example:

- The target price might have been $30 million;

[13] However, benchmarking comparisons of investments are not generally accurate.

- Expected reduction might have been 10% ($3 million);
- The final negotiated price might turn out to be $28 million.

In one case, the difference was claimed to be due to extra tooling not included or estimated in the original plan. If there is no agreement, then Ford will rebid the job at the T- 24 month point. Apparently, there is still enough time for nearly everything except for more complex items such as an engine-block line (which can take additional six months).
Ford is pleased by the new arrangement because now the suppliers function as team members instead of adversaries.

19.9 Lessons Learned

Developing and deploying any new process within a company the size of the Ford Motor Company is a difficult challenge. The following lessons learned reported by Ford managers are based upon the last eighteen months of process development and implementation.

19.9.1 Changing Mind-Sets

Ford's Investment Efficiency process is based upon a fundamental change in mind-set for its product development organization. This mind-set had developed over many years of developing vehicle programs. Such changes in corporate culture do not occur overnight. An analogy is that of a large cruise ship: the ship does not make 90-degree turns, but rather progressively shifts its direction. Therefore, the expectation has to be that the change will be gradual rather than immediate.

19.9.2 Understanding the Need for Change

For changes in mind-set to occur, employees must understand why the change is necessary and feel the need for such a change to occur. In the Ford example, a sense of urgency has pervaded the company for the first time since the early 1980s, driving the company to become investment efficient. The company sees increasing pressures from competitors such as General Motors, Chrysler, and Toyota. Financial results indicate that Ford is lagging behind these competitors when it comes to product development costs. At the same time, the company requires increased capital to enter growth markets overseas while maintaining its market share in mature markets with the introduction of new and innovative products. The overseas growth markets require products that are low cost but still provide outstanding quality and exciting features. For Ford, the need for Investment Efficiency is clear both to its management and employees.

19.9.3 Strengthening Management Support

For such a culture change in a large organization, there must be strong management support and discipline. Management must demonstrate a commitment to the process. In the case of Ford, the establishment of the Investment Efficiency Council to oversee the development of the process and review progress with the platform teams ensures that employees see senior management support. Ford Product Development Management has strengthened its product milestone reviews, allowing no platform team to go through the development gateway with an investment status not at-, or near its investment target. In the past, platform teams would go through these reviews with an inflated status and a promise to get its cost down. Now, these same teams are not allowed to undergo such reviews unless they are on their investment target.

19.9.4 Creating Aligned Objectives

The development of aligned objectives is important to implement a process such as Investment Efficiency. Within Ford, there are several manufacturing divisions and a product development group. If each organization is working to a different set of objectives, then any process will fail.
Ford has begun to break down these organizational 'chimneys' through Ford 2000's use of matrix management. In addition, the Affordable Business Structure targets are the common element to align the business objectives of each organization. Each organization is charged with getting its cost on target with the Affordable Business Structure.

19.10 Discussion Items

The following questions are provided as starting points for discussions on how the Department of Defense can better integrate cost tradeoffs and cost targeting into its acquisition processes and integrated process teams.

1. What is the DoD parallel to the shift in Ford's markets?
2. What advantages are to be gained by Product and Process Compatibility? Do these have parallels in the Defense systems acquisition environment?
3. a. The design rules and metrics at Ford start at a high level but become quite detailed. Ford has created groups of experts to bring detailed manufacturing knowledge to bear early in the development process. What are the implications of these facts on the following:
    - What should a government program office expect of its development and production contractors?
    - What should a government program office expect of itself?

b. Ford uses Product and Process Compatibility workshops to drive the designs of vehicle subsystems to the physical targets established. The purpose of these sessions is to focus the detailed product assumptions with the physical targets established for each system. How would you arrange for a similar process to occur within the product development processes of contractors designing your systems?

4. It was important for Ford to understand its principle cost drivers down to a very detailed level. How can the Department of Defense gain this understanding, given that it must work through its contractors? What contracting mechanisms are available that would cause the contractors to identify the cost drivers at a sufficient level of detail both for themselves and for the government?
5. What special training is required to understand Investment Efficiency and Product and Process Compatibility in the Defense systems context? What training should be provided to bring government and industry's engineers to a level where special groups of experts on costs and investments are no longer required? Does the Department have a role in the education and training of the engineering work force in industry?
6. Reuse of various categories is a fundamental concept at Ford. What are various types of reuse that would benefit defense system acquisitions in costs, investment, and reliability? How could the Department of Defense arrive at a situation where processes and facilities are routinely re-used for missiles, tanks, aircraft, electronics?
7. There are many kinds of costs to try to minimize in tradeoffs. Among them are marginal unit costs, total development and manufacturing costs assuming a given volume, investments over some range of systems (e.g., all missiles), and others. Which of these measures is important in various defense systems acquisition environments? Are there tradeoffs among these different costs?
8. What is the relevance of Ford's use of metrics at its 'gateways' to the Department of Defense's milestone review process? What is the relevance of the composition of Ford's review committee? What is the relevance of the workshop process?
9. How does reuse of physical entities (for example, components) apply in areas with very fast-moving technological progress (for example, electronics)?
10. What management actions, at various levels of management, would be required to implement an investment efficiency strategy such as that described here (a) within your acquisition domain and (b) at a DoD contractor?
11. Ford managers pointedly stated that technology is not the fundamental issue but that *the process* is. What is the distinction they are emphasizing? What process are they referring to? How does that relate to the acquisition area the reader is involved in?

12. Compare the major points of Ford's Product and Process Compatibility and the Department of Defense's IPPD and CAIV initiatives both in theory and current state of practice.
    In particular:
    - Management commitment
    - Motivation
    - Team structures
    - Relevant costs
    - Timely development and treatment of detailed targets
    - Application across projects
    - Consideration of product life-cycle costs
    - Supplier relationships
    - Ability to capture accurate customer requirements

19.11 Glossary of Specific Terms Used in This Chapter

Affordable. Ford uses the term 'affordable' as a function of product development, manufacturing processes, and market demand. See the Affordable Business Structure equation.

Affordable Business Structure. Ford's planning framework for costs that guides development of vehicles that are both affordable to Ford and to its customers. All costs revolve around the price the consumer is willing to pay for a product, and Ford must direct all its effort to producing products that provide the most value for the dollar. (Compare with the 'Classic' product development equation.)

Affordable Business Structure equation. The Affordable Business Structure equation differs from the 'classic' approach to product development by shifting the basis of the product development equation from the company to the consumer. The basis for product development is the amount the consumer is willing to pay (market price). From the market price, Ford deducts its profit target for the product. The fallout of this equation is the affordable cost.

$$MarketPrice - ProfitTarget = AffordableCost$$

(Compare with 'Classic' product development equation.)

Carryover product. A driver of investment, a carryover product is a product component, assembly, or feature 'carried over' to the new model from the prior generation, particularly a part that the customer does not perceive to differentiate value. This enables reusability of the existing manufacturing equipment. 'Classic' product development equation. In the past, Ford's Product Development process for a new vehicle program consisted of developing product

designs; estimating the tooling, facilities, launch, and engineering costs for those designs; adding a profit margin; and thereby determining the revenue target for the product. This relationship is shown in the 'classic' product development equation of

$$ProductCost + ProfitTarget = RevenueTarget(PricetoCustomer)$$

(Compare with the Affordable Business Structure equation.)

Commonality. A driver of investment, commonality is the ability to use product assemblies, features, product attributes, and facilities and tools shared with other products. An example is using a common part across several vehicle lines, thus allowing Ford to avoid spending capital to (1) design unique tooling, (2) engineer the unique part, and (3) build prototypes of that unique part.

Complexity reduction. A driver of investment, complexity reduction reduces the intricacy of a product or manufacturing process. The ability to reduce part complexity will increase commonality of parts across models.

Concurrent engineering. A systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule, and user requirements (Winner et al. , 1989; Nevins et al., 1989).

Cost As an Independent Variable. A Department of Defense acquisition reform initiative. It requires that cost be set in advance of development rather than to emerge as an outcome of development.

Flexible Manufacturing. The ability of manufacturing and product design to respond in the shortest amount of time-to-market changes with products that profitably meet customer needs.

Ford 2000. An omnibus reorganization of Ford's processes.

Ford Product Development System. A cross-functional process that involves all Ford development and manufacturing activities and suppliers. Its goal is to improve quality, cost, and time to market. Created as part of the Ford 2000 reorganization.

Freshening. Ford generally categorizes product changes within a cycle as all-new product, major freshening, and minor freshening, based on the degree

of change. For example, Ford might introduce an all new product offering in the 1997 model year with a twelve-year cycle. It will then plan to conduct a minor freshening on that product after four years, a major freshening after eight years, and will replace the product altogether at the end of the twelve-year cycle.

Gateway. Ford's milestones where program reviews are held and product/process plans are measured on the basis of product features versus costs to achieve. No platform team is permitted to go through the development gateway with an investment status not at or near its investment target.

Generic Product/Process Concepts. One of the four main tools to drive Product and Process Compatibility in its platform teams, Generic Product/Process Concepts provide designs and processes that are 'off the shelf' and that have been optimized for both Product and Manufacturing requirements. These concepts enable Ford to achieve greater commonality across its product lineup and to achieve greater reuse of its manufacturing equipment.

Hardpoints. Specific locations on the vehicle's body pan to be used as attachment and reference points for all tooling, welding, and assembly for the entire vehicle. The approach being adopted across the vehicle industry is that there should be four hardpoints in common for all products in a company's line.

Integrated Product/Process Development (IPPD). A management technique that integrates all acquisition activities starting with requirements definition through production, fielding/deployment, and operational support in order to optimize the design, manufacturing, business, and supportability processes. At the core of IPPD implementation are Integrated Product Teams. Some consider IPPD to be Concurrent Engineering renamed.

Investment Efficiency. The ability to simultaneously minimize investment by Ford and to optimize value for the customer. The goal is to provide the most product for the investment dollar.

Investment Efficiency Metrics. One of the four main tools to drive Product and Process Compatibility in its platform teams, Investment Efficiency Metrics are high-level or detailed measures that indicate whether a product/process design meets project-specific and company-wide cost goals.

Life Cycle Cost Analysis. One of the four main tools to drive Product and Process Compatibility in its platform teams, Life Cycle Cost Analysis at Ford attempts to determine investment and other financial requirements over the lifetime of a product line, including minor and major 'freshenings'. In the Department of Defense, life cycle cost analysis tries to capture costs of

ownership, logistics, operation, and disposal of some type of product.

Manufacturing Design Rules. One of the four main tools to drive Product and Process Compatibility in its platform teams, Manufacturing Design Rules identify critical parameters or constraints that make up the manufacturing professes and are the drivers of facilities and tooling investment.

Micro-engineering. Product teams look at a completed part design to identify tooling opportunities either at the Ford assembly plant or at the vendor manufacturing site. Its goal is to improve cost characteristics of completed component design.

Parts bins. Parts approved for use or re-use in Ford products.

Platform. Three main structural assemblies that make up the underbody of the vehicle: Front End Structure, Front Floorpan, Rear Floorpan.

Product and Process Compatibility. Making sure product designs are compatible with existing manufacturing processes, tooling, designs, and facilities early in the design process.

Reusability. A driver of investment, reusability is using existing prior-model tools, facilities and processes, thus minimizing investment.

Simultaneous Engineering. Manufacturing and Product Design engineers working together during the design of a component.

Target price. During planning and development, the price at which a product is to be sold to the end consumer.

Thrifting. Removing features and options from a product to achieve an investment target.

Total Quality Management. A management initiative in which the driving objective of a company is to continuously improve the quality of processes, intermediate products, and final products. The intent is to drive costs and schedules down through the elimination of waste and rework.

19.12 Acronyms

AMPPE Advanced Manufacturing Pre-Program Engineering

CAD Computer-Assisted Design

CAE Computer-Assisted Engineering
CAIV Cost As an Independent Variable

CAM Computer-Assisted Manufacturing

DoD Department of Defense

FPDS Ford Product Development System

IE&CA Investment Efficiency and Competitive Analysis

IPPD Integrated Product/Process Development

PPC Product and Process Capability

T Target date for beginning mass production (date of Job #1)

References

Bowen, H. K. et al.(1994) The Perpetual Enterprise Machine. Oxford, UK: Oxford University Press, 1994.

Gansler,J. (1993) Affording Defense. Cambridge MA: MIT Press, 1991. (Reprint edition) General Motors. 1993 Annual Report.

Nevins, J. L. et al.(1989) Concurrent Design of Products and Processes: A Strategy for the Next Generation in Manufacturing. New York: McGraw-Hill 1989.

U.S. Department of Defense(2002) For information on Cost As an Independent Variable (CAIV) and other initiatives, see the Defense Acquisition Deskbook, http://www.deskbook.osd.mil. (2002, 2002)

Winner, R.I., Pennell,J.P., Bertrand, H.E. and Slusarczuk M.G.(1989) The Role of Concurrent Engineering in Weapons Systems Acquisition. Alexandria, VA: Institute for Defense Analyses, 1988. Available from the Defense Technical Information Center and the National Technical Information Services as AD-A203 615.

20

THE BUSINESS PROCESS (QUIET) REVOLUTION; TRANSFORMATION TO PROCESS ORGANIZATION

Meir Levi

Interfacing Technologies, Canada

20.1 Introduction

Although research into business processes has been conducted earlier (e.g. IBM (Engelke et al. , 1985) and CIMOSA (AMICE , 1989)), it was Hammer (1990) who first raised the visibility of business processes with the introduction of BPR — Business Process Re-engineering — in the early 90's. In subsequent years, BPR has often been associated with drastic changes and downsizing initiatives, rather than improving practices and resulted in many failed re-engineering mega-projects. The emergence of the Business Process Management (BPM) in the new millennium (post Y2K), has given renewed focus to the process promise and has been a quiet - yet solid - business revolution.

To understand why an entire enterprise would begin instituting process structure and transforming management to business *process* management, we must understand the primary characteristics of the business process, or the *Process Construct* and the benefits brought about by BPM.

The traditional 'Function Enterprise' is the product of the Industrial Revolution in which the guiding principle for organizing enterprises by function is the distribution of work by labor specialization.

In the Process generation, the functional organization of enterprises may not completely disappear, but rather be transformed into the context or the grid for performing processes that bring value to customers.

Technological superiority, innovation, or longevity are no longer what makes or breaks companies — it is how well they are organized to respond to and serve their customers.

The only way to achieve such sustainable customer satisfaction and results is to become a Process organization. Table 20.1 below highlights the important cultural differences between a functional organization and a process-centric one.

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

Table 20.1. Functional vs. Process Enterprise

Enterprise Behaviors	Functional Enterprise	Process-centric Enterprise
Managers Manage	Resources and Work	Customers and Results
Teams Operate	Independently	Collaboratively
Organization Dynamics	Rigid to adapt – Frequent re-org.	Flexible to new demands and self-reorg.
Resources Focus	Meeting job requirements	Best results, Customers
Knowledge Dissemination	Islands of Information	Integrated across the enterprise
Culture	Closed	Open

20.2 Process Classification

Quite often, business processes at different levels are seen as synonymous with workflow, application automation and/or application integration. These 'automated processes' are a sub-set of the overall 'human processes' which make up the process framework of the organization. While selected steps of human processes are traditionally automated using Workflow solutions and/or specifically designed applications, such automation applies to a very specific set of repeatable and frequent processes, sub-processes and activities. Common examples include: 'Call Routing' in the Help Desk process, 'Order Entry and Tracking' in the Order Fulfillment process, Automated Core processes (trading transaction processes, on-line banking, et cetera.). It is important to note that, typically, every automated process is triggered by a human activity or sub-process. The 'Call routing' sub-process is triggered by a call or an email to the help-desk and may commence with a support person responding, which then invokes an automated flow of subsequent activities.
To ensure successful process transformation, both automated and human processes must be managed under the same comprehensive framework (refer Table 20.1). The basic criteria for a successful business process are as follows:

a) visibility to all process stakeholders;
b) value - adding, and
c) efficiency and focus on contributing to customer satisfaction.

Hundreds (and sometimes thousands) of processes make up the process framework of a given enterprise. Classifying them in a manageable top layer (typically consisting of up to 10 top processes) and distinguishing between 'core'

(also called 'identity') and 'support' processes makes the trees visible in the process forest. Identity processes are those that make the enterprise unique in its market space, while support processes are the same from one enterprise to the other (Finance, Admin, HR, and others). Once created, the process hierarchy must be maintained like the enterprise's organizational chart.

Table 20.2. The Business Process Construct

The Enterprise Process Framework:			
Support Processes		Core/Identity Processes	
Finance • Credit Authorizatn. • Budgeting • Auditing	Marketing • PR/Communication • Web Marketing • Lead Generation • Events/Trade Shows	Banking/Finance • Straight Through Processing • Acct Provisioning • Loan/Credit Processing	Services • Outsourcing • Application Development • Logistics
Ops/Logistics • Purchasing • Contracts • Invoicing • Shipping	Sales • Qualification • CRM • Pre-sale • Negotiation & Closing • Channel Mktg.	Energy: • Short Term Trading • Long Term Trading • Strategic Planning	Pharmaceutical • Clinical tests • Drug Submissions • Clinical Research
Human Resrc • Hiring • HR Developmnt • Perform. Evaluation	Legal • Contracts • Policies • Acquisitions	Manufacturing • Product R&D • Product Engineering • Quality Assurance • Production	

An enterprise is a microcosm, with people, behaviors, activities, goals, aspirations and so on, working in a non-isolated environment of market pressures, customers, suppliers, regulatory laws.
What differentiates a successful enterprise from a lagging one is the way it is organised - not just to satisfy its constituents (customers, suppliers, employees and shareholders), but to actually please them with great experiences.
The way to achieve this dimension of satisfaction is to create an environment in which all parties collaborate towards common goals and results.
The business process framework enables:

a) Alignment and Consistency: Process stakeholders gain a clear understanding of their process and align to execute the process in a consistent manner;

b) Execution: A clear process holds its owner accountable for a high degree of execution and maximization of results
c) Optimization: Well-defined processes are easier to improve and optimize.

20.3 Process Management with FirstStep [1] and the EPC

Business Process Management is about availability of critical and up-to-date process information to process managers who are accountable for the process execution and goals. A number of process-modelling and -management solutions supporting several methodologies have been developed over the years. Being a methodology independent enterprise modelling and simulation application, FirstSTEP has been adopted by industry leaders in such diversified sectors as manufacturing, finance, telecom, energy, healthcare, public, and services. Its concept and approach comply with the CIMOSA framework (CIMOSA Association , 1996).
Today, the FirstSTEP family of process modelling products includes two members: FirstSTEP Designer for modelling business processes, performance analysis and simulation, and FirstSTEP Charter for process mapping in MS-Visio. Both modelling environments use XML as a bridge between each other and to the Enterprise Process Center™ – the EPC[2].
The EPC is a web-based (J2EE[3]) knowledge and process portal that enables access to process information (maps, models, documents, applications, process instances, process data) from every desktop inside the enterprise firewall. Optional extranets allow the extension of the EPC to customers and suppliers outside of the enterprise boundaries.
While the FirstSTEP process design environment is evolving to become an integral component of the EPC, the EPC will also support other process meta-models so enterprises can benefit from existing investments in process design work.

20.4 The new Business Process Construct

Perhaps the most concise definition for 'business process' is the one suggested recently by Hammer (2001): "an organized group of related activities that together create a result of value to customers" (refer Fig. 20.1).
Note that distinct activities, performed by a single or multiple functions within a single or multiple enterprises, join together to deliver results to a client.

[1] FirstStep is a trademark of Interfacing Technologies, Inc
[2] for more details on the FirstStep modelling tool also refer Chapter 3 and the Interfacing Technologies web site (at the time of this writing, http://www.interfacing.com.)
[3] Sun's Java 2 platform, Enterprise Edition - at the time of this writing http://java.sun.com/j2ee/

Operating through a process, ensures that focus is kept on everything that makes the process results shine and the process client delighted.

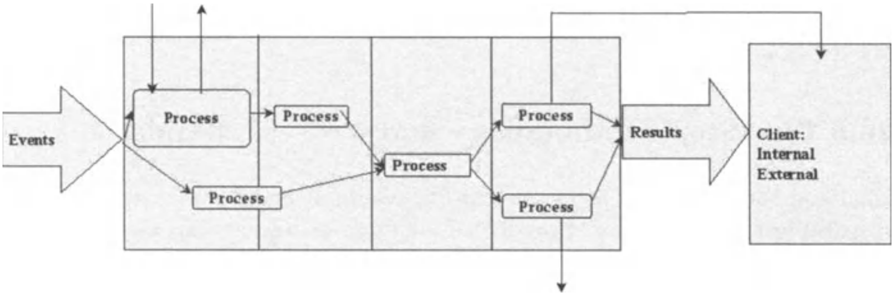


Fig. 20.1. The Business Process Construct

With an enterprise-wide view on processes, a process chart, or 'Process Framework', emerges. Such a framework provides a well-defined structure for the web of processes that traverse the enterprise. All key processes and their interrelations, links to enterprise objects, human resources, assets, information, knowledge and supporting applications ultimately make up the framework. Processes can be defined at many different levels and with various boundaries. To derive the important benefits that processes bring to the enterprise management infrastructure, we need to consider a number of important process characteristics:

- Process Results: These are clear performance targets linked to the organizational strategic objectives and designed to support the mission and the direction of the enterprise.
- Process Boundaries: Boundaries define the scope of the process: its beginning and end-points. Furthermore, they determine the 'touch points' with other processes.
- Process Instantiation: Instances of processes can be:
  - transactions in a repeated transactional process,
  - projects in a project driven business unit, or
  - programs in a service delivery organization.
- Process Client: The ultimate customer who will enjoy the benefits of the process and receive the value generated by all process constituents.
- Process Manager/Owner: This is where the responsibility and accountability for the performance of the process lies.

To create a beneficial *Process Framework* for the organization, one must create the 'top level tier' first to provide a clear vision or structure, and then proceed to subsequent process tiers in a way that can lend itself to a clear and efficient process distribution through the organizational functions.

In constructing the top level tier, it is important to make a distinction between core processes which are sometimes also called 'identity processes' (Hammer , 2001) and 'support processes'. Support processes are typically transactional in nature. Identity processes can accommodate all types of process instantiation. The types that are used depend on the nature of the enterprise.

20.5 FirstStep Methodology and Process Standards

Charting the enterprise processes can be accomplished in many ways —from the simplest diagramming approach to the creation and management of multi-layered and complex maps coupled with various organizational objects and linked to systems, information, networks and other dimensions. Currently, there seems to be a renewed effort to converge Business Process Modelling standards. The Object management Group[4] (OMG) and the Business Process Management Initiative[5] (BPMI) are making efforts to converge and create a common language.

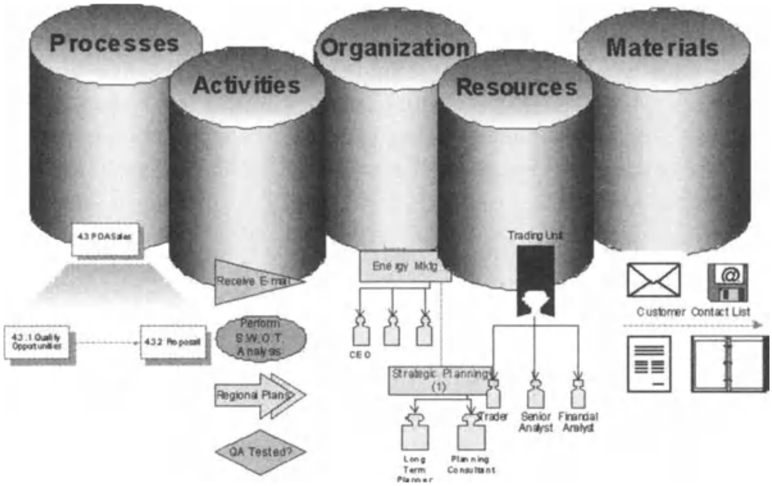


Fig. 20.2. Modelling Language Constructs used in FirstSTEP Methodology

The FirstSTEP modelling class definition has been described in (Levi et al. , 1999). It consists of five classes of objects:

- Processes (with hierarchical layers of sub-processes);

[4] at the time of this writing, http://www.omg.org
[5] at the time of this writing, http://www.bpmi.org

- Activities (the lowest-level work steps, classified in 6 generic groups for clarity and standardization),
- Organizational units (functions, departments);
- Resources (performers of activities, human and non-human – systems, machinery);
- Materials (products, documents, information objects – all entities flowing through, linked to, or referred to the process).

This definition represents a common subset of most evolving standards and thus is designed to adapt to most enterprise needs.
The FirstSTEP methodology (refer Fig. 20.2) focuses first and foremost on the business design needs. By being simple and generic, it can be easily understood by business users at all levels, a pre-condition to the acceptance of process framework and getting the buy-in from all users in the enterprise.
The methodology is evolving to support integration to other meta-models such as represented by evolving standards; UML[6], BPML[7], IDEF[8] and WfMC[9].

20.6 Deploying Process Framework with the Enterprise Process Center

The transition to Process Enterprise takes a concentrated level of effort. It requires attention to many operational, organizational, human and systems/application details. The process may take several months and considerable effort to shape up, depending on the size and state of the enterprise. The rewards however, justify the process and the efforts.
Through working with a number of enterprises, Interfacing Technologies has developed a three step program, called **PI-3** (refer Fig. 20.3), supported by the Enterprise Process Center technology and designed to take an organization through the transition path.
While many enterprises begin the transition without doing so, it is necessary to ensure clarity in the process, especially when it involves everyone in the organization.
Coupled with the FirstSTEP methodology, and supported by both FirstSTEP and the EPC, PI-3 takes advantage of the hierarchical approach to setting the high-level process framework in its first step: the *Process Infrastructure*. The enterprise process framework is created in this step. The framework typically consists of 6 to 10 primary processes at tier 1, with decomposition to multiple

---

[6] Unified Modelling Language, a suite of modelling languages used mainly for software modelling and standardized by the Object Management Group. Currently at http://www.omg.org/uml/

[7] Business Process Modelling Language, a meta-language for the modelling of business processes, currently at http://www.bpml.org

[8] Integrated DEFinition Languages, http://www.idef.com

[9] Workflow Management Coalition, http://www.wfmc.org

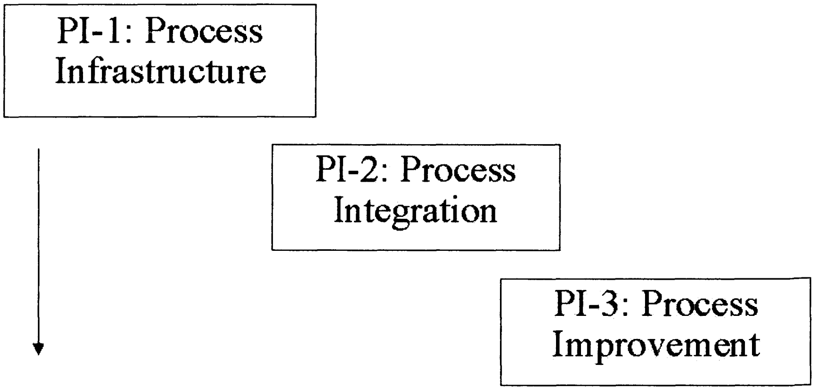


Fig. 20.3. The PI-3 Program

levels of sub processes, organized in a methodical fashion. The EPC, with its hierarchal view (process trees) enable easy access to any layer and process.

Processes have numerous links to one another, and it is in those links that errors in executions often occur. The *Process Reconciliation* is a key step required to ensure clarity in what is expected from one process to support the other. Inter-process collaboration is fundamental element for a solid infrastructure.

The second step in the program focuses on integrating the process framework into the Enterprise's existing systems, applications, content and knowledge. This is the *Process Integration.*

Process knowledge captured within the first two steps in either maps or in related content, creates enthusiasm for the promise of process management (Fig. 20.4).

Users, managers, and external partners can view the processes they own or in which they participate, access critical knowledge and applications, and execute with clear view of the impact of their actions. Process knowledge must be live and incorporated into the evolving business practices.

The Process Framework can now become an integral part of the organization, enabling access to critical data and specific applications directly from the corresponding process step. To complete the integration, applications and systems must be able to feed back critical information to assess and audit the processes.

The third element in ensuring that processes meet and exceed set targets and results are captured in the *Process Improvement* phase (refer Fig. 20.5). Goals must be measurable and verifiable. And then improvement proposals can be implemented with clear ability to assess the incremental results.

Nowadays, the art of business management includes quantified decision making techniques, such as those promoted by various improvement and quality methodologies. The most process oriented one is Six Sigma (Pande et al. ,

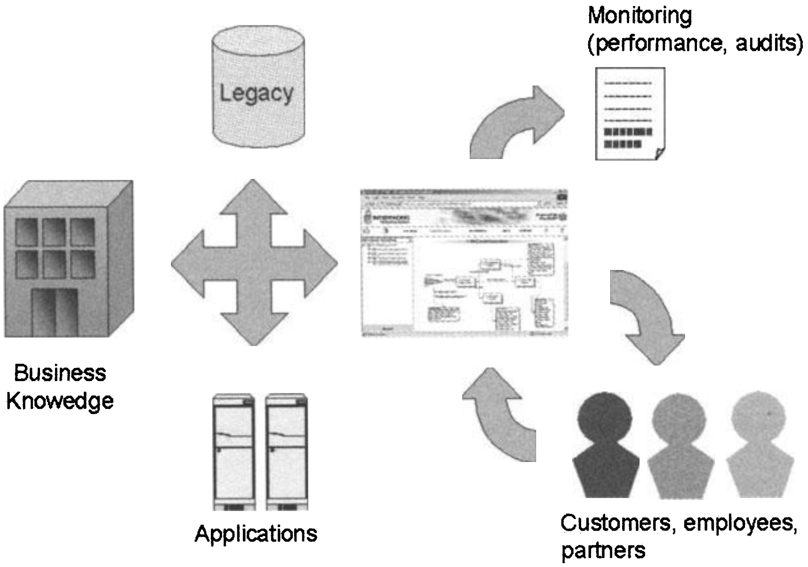


Fig. 20.4. Process Knowledge

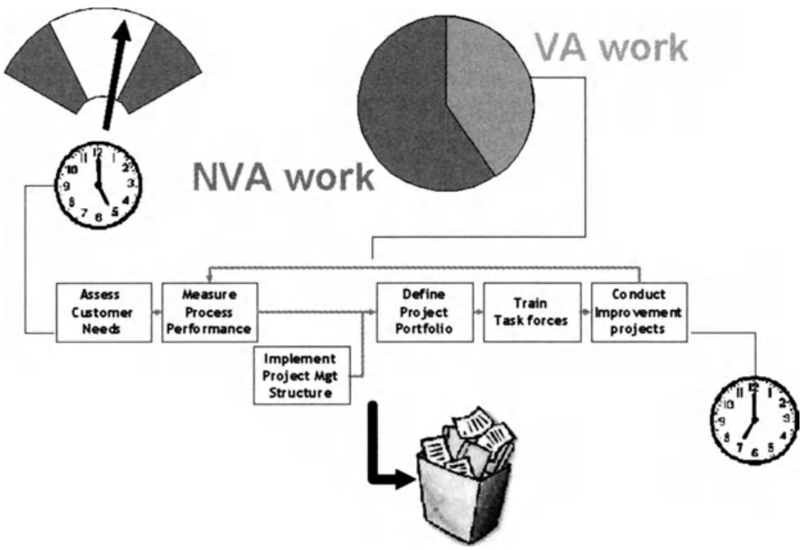


Fig. 20.5. Process Improvement

2002). Others target both organizational and process key measures; the most employed one is the Balanced Score Card (Kaplan et al. , 1996). The goal is to ensure that the standardized processes deliver results that meet customers' requirements in the most effective way, and if they do not, to act on the real cause of process waste.
Integrated with the other two dimensions of PI-3, the process improvement dimension will enable the process-led organization:

- to maximize customer satisfaction
- to eliminate ambiguity, making critical decisions based on measured facts
- to document priorities for processes improvements
- to eliminate root causes of poor performance, reducing non-value added (NVA) work
- to maximize sustainable business results by focusing on added value (VA) work.

20.7 Initial Case Report

Since the first release of FirstSTEP in late 1994, Interfacing Technologies has worked with hundreds of enterprises, initially to support BPR initiatives. Focus has shifted in the last few years to BPM and the transformation to process organization. In this paper we report notable initial results from a significant project with a leading US based energy generation and trading organization. In 2001, this enterprise decided to become a process driven enterprise. To support them in this mission, they have turned to Interfacing Technologies' FirstSTEP and EPC technology.
Key expectations for drastic impacts were:

1. Making business process knowledge available to team members across the organization, thus adding value through cross-functional knowledge transfer;
2. Increasing operating efficiency with the following mechanisms:
   - continuously challenging and improving how they do business;
   - clearly defining accountability;
   - defining IT needs through process workflow;
   - enhancing auditing capabilities;
   - enhancing performance management capabilities;
   - providing flexibility in a growth environment;
3. Streamlining the process of integration in a merger or acquisition;
4. Self-reorganization through dynamic process changes. Changes in process ownership, boundaries and definition automatically imply new roles and responsibilities, hence accomplishing in a more natural manner what was traditionally done through functional reorganization, often viewed as a drastic, costly and unpleasant process.

5. Improving HR management. Skill gaps are identified through resource needs of processes—workforce planning, a much quicker process of hiring and reallocation of resources.

The project started by outlining and getting consensus on the high level process framework structured around 10 primary core and support processes. Process Leaders were appointed to each of the process areas. This was the start of the Process Infrastructure phase, during which hundreds of processes were mapped through a methodic process involving process owners and process stakeholders.
In addition to a simple but effective process review and validation, the Process Reconciliation phase was a very important step in completing the process framework. In this phase, cross functional and cross process teams identified key process touch points where critical interfaces between distinct process areas occur.
The project is supported by the EPC which enables a single-point access to all of the key processes and their related content. This includes:

- Business Process Flows, Maps and Diagrams
- Application and System Linkages / Integration
- Process Content Integration; documents, spreadsheets, websites, presentations, procedures, templates
- Process Ownership and Accountability

While the project is in the middle of the transformation process, with the ultimate benefits yet to be realized, promising observations and results have already been reported:

- the entire Company is 'process aware';
- process 'members' share information and are enthusiastic about process definitions and executions;
- one place has been created for employees to go for 'Process Knowledge';
- 'applications' fall into the 'Big Picture';
- operations are gaining higher level of consistency expected to yield direct impact on bottom-line.

To achieve the initial results however, major challenges had to be overcome (refer Table 20.3).
The *Infrastructure* phase of the project is near completion at the time of writing this paper. The *Integration* phase is being executed with process content and initial applications being integrated. The process improvement initiative, which will tie organizational and process metrics to the Framework, will be launched next. With the ability to set, measure, monitor and manage process related Key Performance Indicators (KPIs), process ownership and accountability will be realized. This will enable true management by process results, in order to maximize the organization output and bottom line.

Table 20.3. Challenges in the project

Typical Challenges	Steps taken
Resistance to Process Change	Regular Process Leader and Process Manager meetings, short/frequent education sessions
Learning new ways and practices	Get everyone involved in capturing and reviewing Process information
Assuming responsibility for results	Clear KPI based incentive programs
"No time ..."	Prioritization
Doubts and all of the above	"Commitment from the very top – the CEO"

20.8 Summary and Conclusion

It is only now that the biggest impact of process framework technology has the opportunity for significant realization. While many enterprises have embarked on one level of BPM initiative or another, only those who completely change their culture to become Process Enterprises will gain sustainable rewards. Initial results reported in this paper illustrate pioneering work of one of the first enterprises to embrace process culture, and the initial results are significant.

Challenges and threats to such deployments are very real and the deployment teams had to continuously find ways to meet the challenges. Without a doubt, the most critical success factor was the complete and unconditional directive from the very top - the CEO - to see the transformation process to its completion.

Future work, already underway, will see the smooth integration of the high-level Process Framework to the enterprise application and data layers to increase the ability to automate and integrate. The general BPM market is expected to experience the merging of multiple methodologies and meta-models to support similar initiatives.

References

AMICE (Eds.) (1989). Open Systems Architecture for CIM, Springer-Verlag

CIMOSA Association (1996). CIMOSA, Open System Architecture for CIM, Technical Baseline, Version 3.2, Release, private publication.

Engelke, H., Grotrian, J., Scheuing, C., Schmackpfeffer, A., Schwarz, W. Solf and B. Toman, J. (1985). Integrated Manufacturing Modelling System, IBM Journ. of Research and Development, 29(4).

Hammer, M. (2001). The Agenda: What Every Business Must Do to Dominate the Decade, pp 51 – 60, Crown Business Publishing Gr.

Hammer, M. (1990), Re-engineering Work: Don't Automate, Obliterate, Harvard Business Review, pp 104-112, July-August.

Kaplan, R.S., Norton, D.P. (2001), The Balanced Scorecard, Translating Strategy into Action, Harvard Business School Press, Boston

Keen, Peter G.W. (1997), The Process Edge: Creating Value Where It Counts, Harvard Business School Press,

Levi, M., Klapsis, H. and Marios, P. (1999), FirstSTEP Process Modeller, a CIMOSA Compliant Modelling Tool, special Issue Computers in Industry, 40(2-3)

Pande, Peter S. Newman, Robert P. Cavanagh, Roland R. (2002), The Six Sigma Way; An Implementation Guide for Process Improvement Teams", The McGraw-Hill Companies, Inc.

21

FARLEY REMOTE OPERATIONS SUPPORT SYSTEM

John Mo

CSIRO *- Manufacturing and Infrastructure Technology (Australia)

21.1 Introduction

Farley Cutting Systems Australia (FCSA) is a CNC machine manufacturer in Australia. The company has customers all over the world and hence needs to provide effective and responsive remote support to customers in the use, maintenance and troubleshooting of their equipment. As the market for CNC machines expands, these needs will increase extensively in the next few years so that the current support processes will no longer be economically viable. In addition, the company desires to capture the extensive expert knowledge within and outside the company concerning their CNC machines, especially as it pertains to problem diagnosis. However, there is no ready means of gathering and storing this knowledge, and hence no way to exploit it for remote customer support.

The emergence of high bandwidth telecommunication networks on which large amounts of data can be transmitted electronically have provided the potential to support customers over long distances. Control systems manufacturers have been successfully providing diagnostic services via telephone networks to locate faulty boards and to place replacement orders. Engine diagnostics and operational support for ships is being offered via satellite connections. Electronic distribution of documents for aircraft maintenance have been developed and offered by aerospace companies via special intranets where the broadband services are delivered by specialised network provider companies. All of these methods have proved the concept that the electronic distribution of documents and services are feasible but the costs associated with such information technology tools are too high for small-medium companies where the margins are critical. The use of the Internet offers much more cost effective solutions.

* Commonwealth Scientific and Industrial Research Organisation, http://www.csiro.au

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

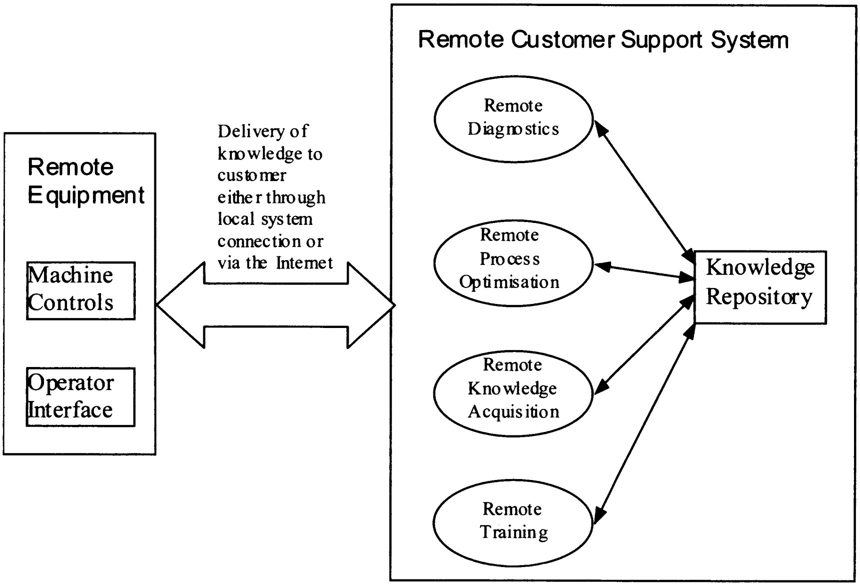


Fig. 21.1. System architecture for Farley remote operations support system

From these observations, the remote diagnostics system is built around a knowledge repository consisting of the field and engineering knowledge of the industry partner and their customers, as shown in the system architecture in Fig. 21.1. Four user interfaces (Remote Machine Diagnostics, Remote Process Optimisation, Remote Knowledge Acquisition and Remote Operator Training), which interact with the user at remote location, interrogate the Knowledge Repository to provide appropriate information to the user. In this way, all functions can easily be kept up-to-date (Mo and Menzel , 1998a).

21.2 Knowledge Representation and Process Modelling

The basic problem of the project had been the creation of a Knowledge Repository. The term 'knowledge' in this context means the understanding of the whole customer support process, including the way the process in question should be carried out, the facts collected in the process and the inference activities which guide the customer to make decisions at critical nodes of the process.

The creation of the Knowledge Repository started with a series of interviews with the experts at Farley in order to master an understanding of the underlying logic that the human expert uses to provide advice in different customer support situations. A number of objectives are identified from these interviews and the characteristics of the domain knowledge are classified in Table 21.1.

Table 21.1. Knowledge Representation and Domain Characteristics

Problem	**Description**	**Knowledge Representation Characteristics**
Data communication	Troubleshooting of communication channel connection between host computer and CNC machine	Standard one dimensional stepwise diagnostic procedure with discrete answers at various decision points
Correction of machine setting	Advise customer on the cutting parameters which can normally produce good machining results	Large amount of experimental and field quality vs. machining parameters to be organised in easily accessible database
Assessment of machining quality	Evaluate machining quality based on machining parameters and advise methods of achieving desired machining quality	Rules are captured from experts in the field and experienced operators to determine if an adjustment of machine parameters can further improve machining quality
Motion system diagnosis	Check whether the source of error originates from errors of the motion control system	On the basis of preliminary machining quality test runs, perform controlled experiments using different machine settings to exaggerate the malfunction of the motion control hardware, if a problem exists
Performance optimisation	Request for an optimised machine setting specific to the product	A trend of the machine's performance is logged and compared with typical industrial operations. A procedure to guide the user to accumulate information about the trend of the machine is required.

It can be seen from the table that the customer support objectives have different characteristics. These problems are normally tackled by different techniques. The solutions and the presentation formats are therefore developed using different skills. For example, problems the causes of which can be isolated by a cause elimination process can be tackled by a diagnostic tree method (Juricic et al. , 1997). This technique is therefore suitable for the trouble shooting of data communication in Table 21.1. Other support objectives, which require some kind of learning capability as the system changes, can use neural network (Chen et al. , 1996; Huang and Wang , 1996) or case base reasoning techniques (Montazemi and Gupta , 1996). For systems with

many interacting components, an object-oriented technique is more appropriate (Batanov and Cheng , 1995).
When structured engineering information is involved, more sophisticated technologies can be used. Patel and Kamrani (1996) have developed an intelligent system for assisting maintenance of automated systems using a combination of group technology and decision tree approaches. Karsai et al. (1996) has built a model-based environment to customise system functionality for specific industrial applications, such as chemical plant diagnosis. For supporting processes with sequencing, duration and synchronisation characteristics, timed Petri Nets provide a convenient modelling language to capture the process in terms of transitions and events (Srinivasan and Jafari , 1993). In real time applications, a combination of on-line system design documents and a rule-based system is required (Combacau and Courvoisier , 1991).
Literature offers a wide variety of methods that can be employed to acquire knowledge from- and providing expert advice to the customer (Guida et al. , 1992; McFarlane et al , 1995). However, in all of these methods the format for supporting training activities also takes a different form (Batty and Kamel , 1995). This means that for the purpose of providing a multi-purpose remote customer support system, the differences among these techniques tend to create isolated sub-systems which do not share a common knowledge base and each application developed for a specific diagnostic purpose has a different look-and-feel (Shankararaman and Lee , 1994). This is a critical issue in operations support management where the need to provide continual support to the customer over a long period of time after the delivery of equipment is an essential element of success. It was concluded that there is a need for a pragmatic approach to create a uniform framework for all knowledge representation. Such a framework can provide a dynamic and flexible method of maintaining the Knowledge Repository and a consistent unified set of the application modules.
Boy (Boy , 1987)has defined five levels of autonomy of the Knowledge Base Systems (KBS) interacting both with a human and a machine system using situation and knowledge blocks. At lower levels of the autonomy scale, the situational part is very small and the analytical part (knowledge) is very large. As the level of autonomy increases, the need to incorporate knowledge and recognition of situations increases, so as to be able to perform a competent and relevant interaction. To elucidate the requirements, during discussions with Farley, the customer support process has been defined as essentially a dialog between the customer and an expert. The customer requests information of a certain general type, e.g., how to trouble-shoot a communication problem between the host computer and the cutting machine. Through a series of questions guided by the answers given by the customer, the expert gradually focuses on the specific nature of the problem, and is then able to suggest actions that are likely to solve it. In the case of evaluating the cut quality, the customer describes over the phone the results of the cut and the parameters used in the cutting operation. The expert then makes a judgement of the

results as compared to his or her experience. The answer may be a suggestion for a change of cutting parameters or a conclusion of some process fault. This suggests that a level 2 system on Boy's autonomy scale is appropriate to model the general *process* involving questions, answers, and actions that experts follow to help their customers.

To capture and maintain this process in a computer requires some intuitive way of *modelling* the process. The fundamental elements of data models are classes and relations. Modelling a process, by contrast, requires a framework for representing the activities involved in the process and the temporal constraints (roughly, the relations of temporal precedence) that determine the structure of the process. In this project, the process modelling language *IDEF3* was used (Mayer et al. , 1992; Menzel and Mayer , 1998).

21.3 Fundamentals of IDEF3

IDEF3 is loosely based upon *situation theory* (Menzel and Mayer , 1996). According to situation theory, the world contains several kinds of things: objects, situations, activities, processes, and process activations. Objects, or individuals, are simply things like people, numbers, NC machines etc. that are not situations, i.e., concrete or abstract objects that have properties and stand in relations to other objects *in* situations.

A situation is a real world *event* across some portion of space and time. For example, the meeting which the author of this paper attended between 08:00 and 08:30 in Melbourne yesterday is a typical situation. This situation comprises some set of objects (the attendees) that have properties and stand in certain relations to each other in the situation. The most fundamental kind of information in situation theory is known as a *basic infon.* Such infons are of the form 'object a has property p' – written og$a$) in IDEF3's underlying 'elaboration language' – and 'objects a_1 ..., a_n stand in the relation r' –– written oga_1 ..., a_n). Boolean and quantified combinations of infons are built up from these.

In most physical systems one observes multiple occurrences of situations that are similar in some respect. In such cases, the similar situations are said to be of the same *type.* Situation types are thus general, repeatable patterns that can be exhibited by many different specific situations. The notion of an *activity* in IDEF3 is identified with an instance of a situation type.

The importance of types in the context of process modelling is that the semantic content (the meaning) of any process model involves almost exclusively types. A typical process is best thought of as a structured collection of activities related to one another by the temporal constraints to determine how the process can be instantiated.

To represent processes intuitively, IDEF3 introduces several basic graphical symbols. For our purposes here we use a simple procedure constraint between

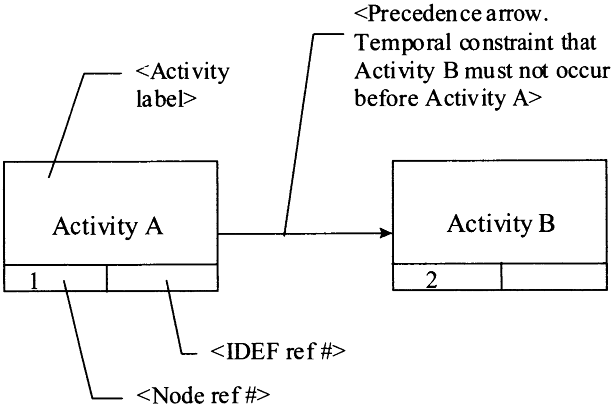


Fig. 21.2. Basic precedence syntax in IDEF3

two activities in a process to illustrate the meanings of the symbols in IDEF3 schematics (refer Fig. 21.2).

The activity instances, i.e., situations, in an activation must occur within a single, finite, interval of time that begins when the first instance in the activation begins, and ends when the last instance ends. To determine whether a given collection of situations is an activation of a given process, it is useful to plot the general pattern of their occurrence over such an interval (refer Fig. 21.3).

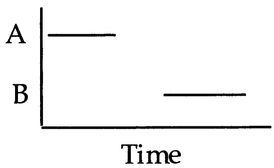


Fig. 21.3. An activation plot for the process in Fig. 21.2

IDEF3 diagrams (or schemata) are type-level descriptions of complex processes Such processes are rarely linear. Typically, processes involve any or all of four general sorts of *branch points*:

- Points at which a process diverges into multiple parallel subprocesses;
- Points at which a process diverges into multiple (possibly nonexclusive) alternative subprocesses;
- Points at which multiple parallel subprocesses converge into a single "thread;" and

- Points at which multiple alternative subprocesses in the process converge into a single thread.

There are four general types of *junctions* to express the four general sorts of branch points. The first two sorts are expressed by *fan-out* junctions: *Conjunctive* fan-out junctions represent points of divergence involving multiple parallel subprocesses, while *disjunctive* fan-out junctions represent points of divergence involving multiple alternative subprocesses. The last two sorts of branch points are expressed by *fan-in* junctions: *conjunctive* fan-in junctions represent points of convergence involving multiple parallel subproceses, while *disjunctive* fan-in junctions represent points of convergence involving multiple alternative subprocesses. Conjunctive junctions are AND type, indicated by '&'. Disjunctive junctions may be either OR (denoted by 'O') and XOR (denoted by 'X').
To illustrate the meaning of these junctions in IDEF3 schematics, Fig. 21.4 shows a model of a simple test process for the communication port of a CNC machine.

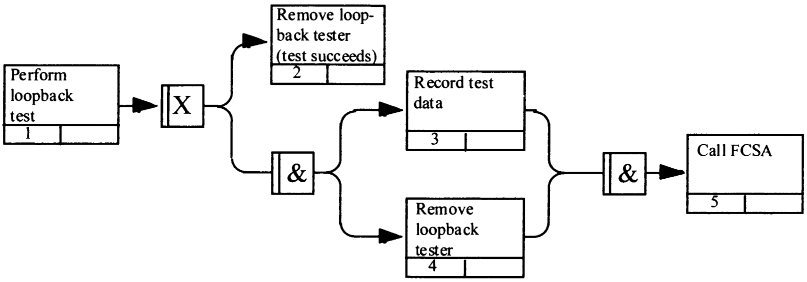


Fig. 21.4. Simple loopback test proces

In this process, following a loopback test, one either simply removes the loopback tester and quits because the test has succeeded (thereby showing, say, that a suggested repair of the system has corrected an earlier problem), or else one records the test data and removes the loopback tester, and then calls FCSA for further assistance. An activation plot is the plot of a possible activation of this process if its structure matches that of the process. Note that there are therefore a number of activation plots corresponding to possible activations of this process, as illustrated in Fig. 21.5.

21.4 The Role of IDEF3 in Providing the Solution

Having explained the fundamentals of IDEF3, we can now discuss the use of process modelling to provide the basis of a solution for remote customer

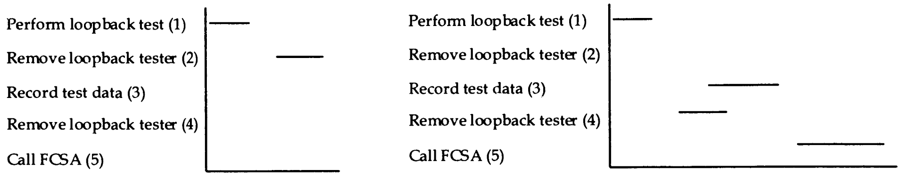


Fig. 21.5. Valid activation plots for loopback test process.

support objectives. The aim for this discussion is to review the appropriateness of the modelling tool and how it can satisfy the knowledge representation requirements of this project.

21.4.1 General Tool for Capturing Process Knowledge

It has been shown that there are many kinds of knowledge required in customer support. E.g., in the case of supporting the diagnosis of a communication link, two kinds of knowledge are required. The first kind of knowledge consists of a description of a specific, well-defined task: installing a loopback tester, running a program, cleaning a cable end, etc. The second kind of knowledge is represented by the temporal and logical *pattern* which these tasks must exhibit in order for them to collectively achieve the given end, i.e. a successful diagnosis. A process modelling language can capture and distinguish both types of knowledge. The knowledge of individual tasks is captured as the content of particular activity boxes in the model of the dialogue process with the customer, while the temporal and logical pattern is captured in the junctions and precedence relations ('arrows') of the model. Such knowledge, of course, is as critical to a successful run of the procedure as the knowledge of the individual tasks.

21.4.2 Re-use of Known Processes

A process often involves multiple occurrences of the same task or activity. However, the *position* of a task in the process is crucial for understanding what to do next. For example, one activity in the communications diagnostic process involves the running of a loopback test program. This activity is performed in the diagnostic procedure whenever the loopback is moved from one point to another. Consequently, which particular task follows a loopback test in a given instance of the communications diagnostic process depends on the process context in which the particular loopback test is performed. Thus, information about what to do *next* in the process cannot be contained within any rule or description of the loopback test proper – unless the description is specialized so as to distinguish between the different appearances of the task in the process. In IDEF3, the structure of the process model captures precisely such a distinction between the task and its context, while at the same time

allows the re-use of the same task definition. This simplifies the application module development through reusing task definitions as components.

21.4.3 Visualization of Knowledge

Visualization of the process is just as critical to Farley as to its customers. For it is these people who must develop and maintain the system. To develop the system for such a complex process though the use of conventional expert system languages has severe limitations, because that technology depends on the involved people being thoroughly trained in AI techniques. IDEF3's graphical representation on the other hand can be easily understood by people. The customer support process can be described explicitly as a 'process map' so that in-house experts in Farley can construct and maintain it themselves. Once this map is created, it can be refined and continuously improved in collaboration with other experts and/or users.

21.4.4 Ease of Knowledge Maintenance

Another crucial task for which a process model is the *maintenance* of the remote customer system. Customer support processes are very fluid: changes in machine design, developments in testing procedures, etc. all lead to frequent changes in the diagnostic process, both with respect to the content of particular activities and to their overall structure. A critical requirement for this project was that the Web-based system should be easily maintainable by the machine supplier. To achieve this, it was essential that Farley personnel would not be required to maintain the system by direct manipulation of HTML files. The solution was to use a process modelling tool (supporting IDEF3) followed bu the export of the IDEF3 models to HTML format.

21.4.5 Integration of Knowledge Components

So far we have only looked at the explicit procedural characteristics of some of the remote customer support process. The general process / dialog with the customer in remote access circumstances can be represented by a high level IDEF3 model as shown in Fig. 21.6.
The step 'Apply appropriate knowledge' includes suitable decomposition of the process into its components. Depending on the type of problem, the sophistication of the decomposition may vary. In the troubleshooting of data communication, for example, the steps of reasoning in the diagnostic procedure can be represented straightforwardly as an IDEF3 sub-model. In contrast, the evaluation of cut quality is far more complicated and subtle, involving the consideration of complex interactions between the machine parameters and the output quality factors. The decomposition of this process therefore defines a link to an underlying rule-based expert system. This approach has the

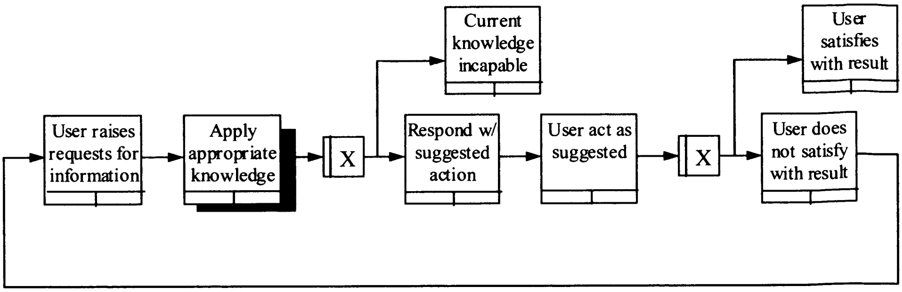


Fig. 21.6. Remote customer dialog process

advantage that a significant part of the reasoning and fact acquisition process can be represented explicitly by IDEF3, and as a result the complexity of the rule-based system can be reduced (given that in this manner the rules can be developed for a given fixed context, rather then in a context independent fashion).
In this respect, the underlying rule-based system forms an essential component of the complete customer support system. It was important to have a uniform approach across all the components of the total system, i.e. a process model was required for each component, regardless of its complexity.

21.5 The System Framework

The approach includes the use of the following technologies: World Wide Web (WWW), hypertext markup language (HTML), interactive forms, graphics, video/audio, artificial intelligence, process modelling by IDEF3, object oriented technology programming(Java). The system, 'Framework for the Integration of Remote Support Technologies', also known as FIRST, consists of 4 blocks as shown in Fig. 21.7 (Mo and Nemes , 1998b). While FIRST was originally implemented only on the Unix platform, a new version of FIRST was later made available for the WINDOWS NT platform (Mo and Law , 2000).

21.5.1 Modelling Tool

The essence of the approach has been the use of IDEF3 process modelling tools to capture the knowledge of the expert in an explicit fashion. The logistics and procedures of solving problems for customers are represented as a series of process charts which can be easily understood by all involved personnel. Knowledge blocks were defined which included 'problem solvers' that are not divisible but could be called upon to provide answers in defined particular context. Once the model was captured, the modelling tool was used to convert the model into an external format that could then be translated into the final deliverable accessible through a Web browser.

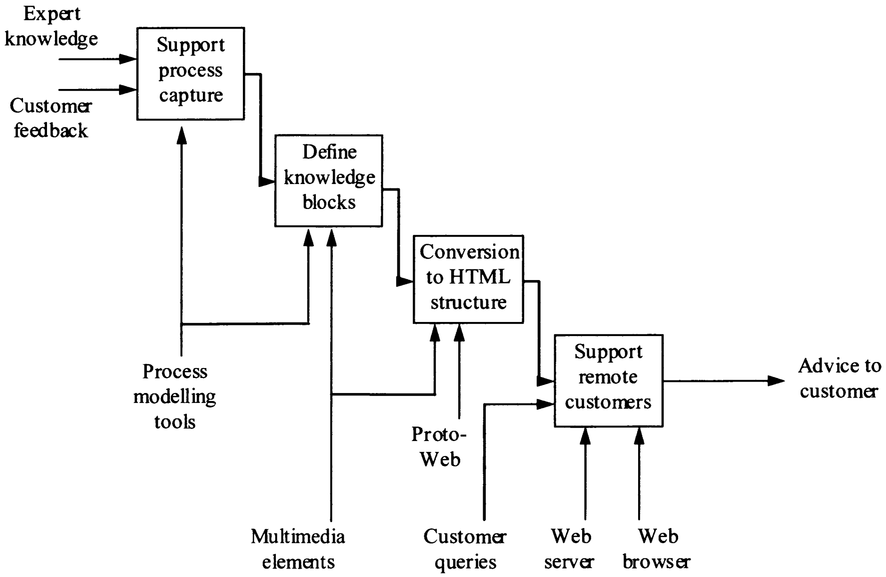


Fig. 21.7. FIRST system framework

IDEF3 process models were created using ProSim$^{\mathrm{TM}}$by Knowledge Based Systems, Inc. (KBSI , 1995). ProSim is a modelling tool that provides an IDEF3 front end for the construction of simulation models. The tool provides a simple intuitive graphical user interface for the editing of IDEF3 process models.

21.5.2 User Interface

In order to deliver customer support information remotely, the project had to develop a method that transforms the process model into an actual end user application. Several requirements were specified for the user interface:

- The system should be easily accessible through the Internet and other telecommunications channels
- Preferably, the system should use free or low cost application front ends
- The same diagnostics system should be viewable on any platform including PCs
- The system should support multimedia applications

The obvious choice was to generate the system as a system of Web pages in HTML, which could also be supplemented with interactive components using Common Gateway Interface (CGI) scripts, Java applets (or any other server side processing technology), so that an interactive user interface can be created.

Using Web browsers as the primary means of accessing the knowledge base, the Remote Machine Diagnostics System is created from a series of background processes from the Knowledge Repository. Using ProSim, the diagnostics process model is exported as a structured text file, which is then converted by a program developed in this project to a set of HTML files, one file for each node in the model. The user can then be guided through the diagnostic procedure using a Web browser (refer Fig. 21.8 for a typical Web page of the system). With the conversion program, whenever there is a need to change the structure of the diagnostics system, Farley needs to update the process model using ProSim or add other knowledge representation modules such as a rule-based system. The modified knowledge base is then automatically re-generated by the conversion program.
In order to allow interactive browsing and manipulation of a process model, a Java program is incorporated as an applet into every HTML page. The applet is activated by a 'Web Map' button located on each page and activates this Java program in a separate window.

21.5.3 Directory Structure

While the system of HTML pages is generated from an IDEF3 model, it has to be linked to the other components of the system. In FIRST, this is achieved by a pre-defined directory structure.
This directory structure is shown in Fig. 21.9.
The 'pics' directory has a 'common' sub-directory and 'models' sub-directories. All pictures, video, audio files are stored in this directory. The 'common' sub-directory contains pictures, video and audio files which are common to all models.
The 'models' sub-directories contain similar types of files but are used separately for each individual model. If the model is removed from the server, the relevant sub-directory will be removed as well.
The 'clips' directory contains all the executables which are activated by CGI-bin scripts. E.g., it contains the common knowledge bases for all users. For knowledge and information specific to individual users, user sub-directories are created under 'clips' and contain data files specific to the user(working directories of each user session). Examples of the data files include .bin, .clp, .dat, .log files. Some of these files are created by the executables during run time. This arrangement separates the data files of individual users so that information can be specific to the need of individual customers.

21.5.4 The Knowledge Based Diagnostics Module

An important part of the Knowledge Repository is the knowledge based diagnostics module, which provides expert advice to the user for the problems of 'correction of machine setting' and 'assessment of machining quality'. Analysis of the knowledge representation models shows that these two problems involve

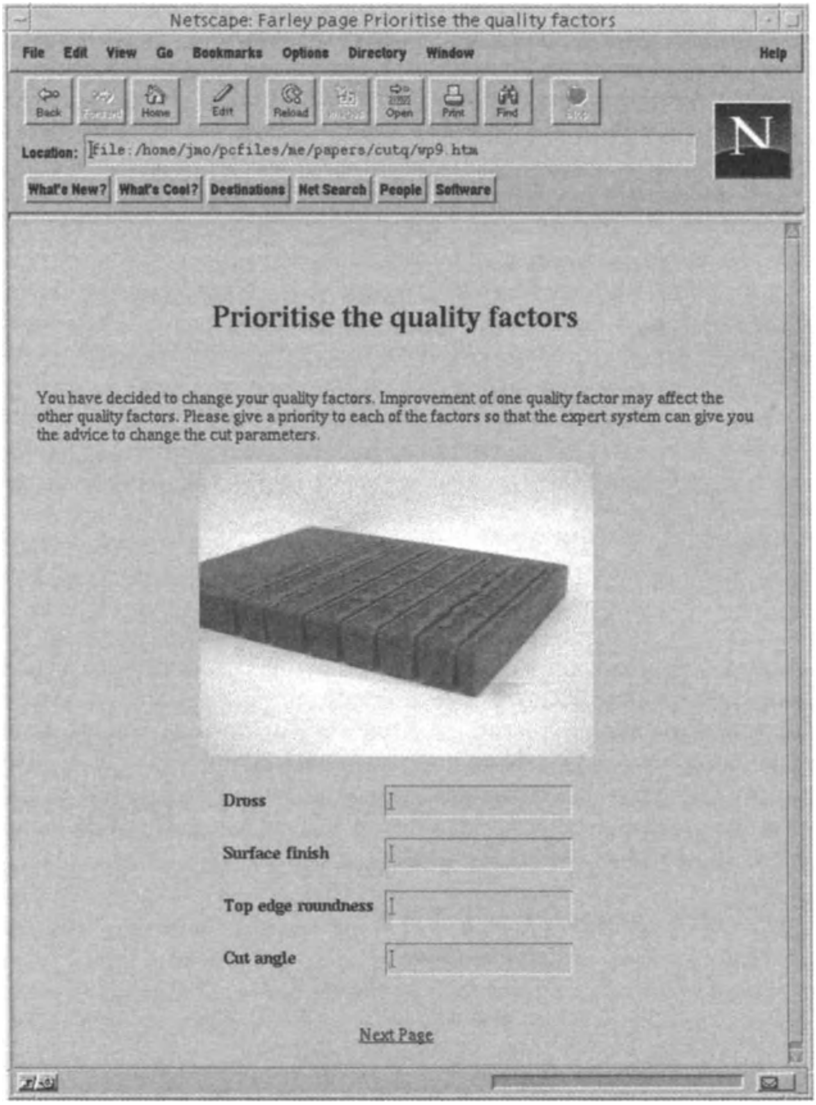


Fig. 21.8. Sample HTML page on the Web browser

a number of inter-related effects between the outcomes of the machining process and machine parameter settings. This type of knowledge is difficult to be represented by the decision tree approach adopted for the other problems such as 'data communication'.

In assessing the machining quality of a machining process, an expert in the knowledge domain would already have a pre-existing experience to support his or her reasoning process. Facts and rules are static in this knowledge domain.

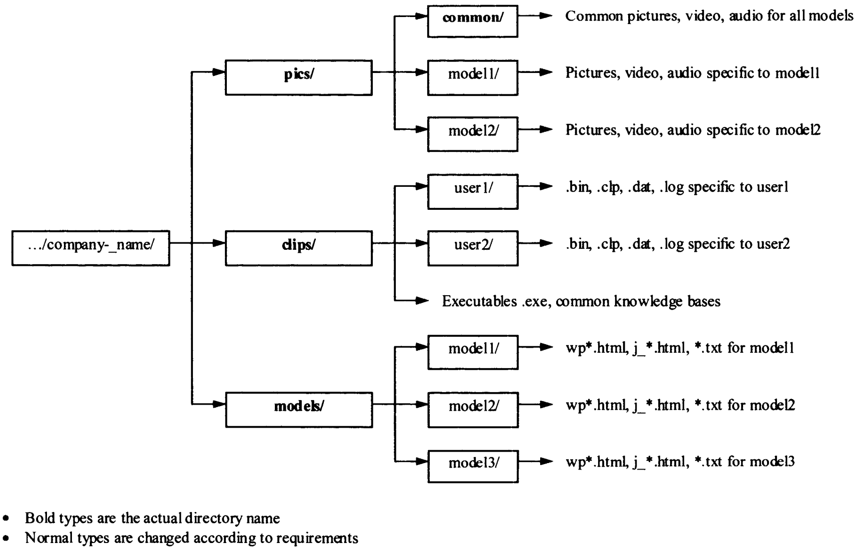


Fig. 21.9. Directory structure to support FIRST

For example, 'the machining quality of a machining operation with a speed of 3 m/min. and a cutter size of 5 mm. is good'.

During actual machining operations, however, due to other conditions (such as input materials), these machining parameters may not be set exactly according to these predefined nominal values. Additional reasoning ability is therefore provided by a number of rules. A typical rule in the rule base system for CNC machine diagnosis is presented below:

```
IF ((surface_finish > 1) &
      //input condition
            (speed > MAX_RATE) &
            (desired_quality = GRADE_I) &
            (quality not TYPICAL))
      //asserted fact
THEN (reduce speed by 1 unit)
```

Hence, the procedure of an expert in the diagnosis of CNC machine normally starts with a series of investigations about the input conditions of the process. This task is the starting activity 'Enter machine information' in the IDEF3 model.

Knowledge-based systems are generally suitable for this type of problems, i.e. which are not well-structured but can be represented by 'IF-THEN' reasoning rules. In this project, the expert system shell CLIPS was used (Giarratano , 1993) to encode machining knowledge. When all the background information

is entered, this information can then be processed by the rule based system to produce an advice to the user. Technically, after the rules base is fully tested, it is compiled into a C program executable, which is referenced in the 'Determine solution' block of the IDEF3 model.

21.6 Conclusion

The approach has received wide attention from industry. Farley planned to commercialise the system as an upgrade to their machines in the field. Another machine manufacturer in Australia was also interested in the technology and started a project to include more sophisticated diagnostics algorithms in their remote operations support system. It was estimated that 80% of existing customer queries could be supported by the new method. Furthermore, a new application of FIRST has been found, in the area of global quality control system development within the aerospace industry (Mo , 2000). The methodology has provided a quick and easy way of maintaining a consistent advanced quality control system for companies working together in the complex supply chain network of aircraft manufacturers. New applications using similar approaches have initiated on-going researches in the design and modelling of remote customer support systems (Kamio et al. , 2002).

References

Mo J.P.T., Menzel C. (1998a). An integrated process model driven knowledge based system for remote customer support, Computers in Industry, Vol.37, 1988, pp.171-183

Juricic D., Znidarsic A., Fussel D. "Generation of Diagnostic Trees by Means of Simplified Process Models and Machine Learning", Engineering Applications of Artificial Intelligence, Vol.10, No.1, February, 1997, pp.15-29

Chen Y.B., Li X., Orady E. (1996) Integrated Diagnosis Using Information-Gain-Weighted Radial Basis Function Neural Networks, Computers and Industrial Engineering, Vol.30, No.2, pp.243-255

Huang H.H., Wang H.P.B. (1996) Machine Fault Diagnostics Using a Transputer Network, Computers and Industrial Engineering, Vol.30, No.2, pp.269-281

Montazemi A.R., Gupta K.M. (1996) An Adaptive Agent for Case Description in Diagnostic CBR Systems, Computers in Industry, Vol.29, pp.209-224

Batanov D.N., Cheng Z., (1995) An Object-oriented Expert System for Fault Diagnosis in the Ethylene Distillation Process, Computers in Industry, Vol.27, pp.237-249

Patel S.A., Kamrani A.K.K. (1996) Intelligent Decision Support System for Diagnosis and Maintenance of Automated Systems, Computers and Industrial Engineering, Vol.30, No.2, pp.293-319

Karsal G., Padalkar S., Franke H., Sztipanovits J., DeCaria F. (1996) A Practical Method for Creating Plant Diagnostics Applications, Integrated Computer-Aided Engineering, Vol.3, No.4, pp.291-304

Srinivasan V.S., Jafari M.A. (1993) Fault Detection/Monitoring Using Time Petri Nets, IEEE Trans. on Systems, Man and Cybernetics, Vol.23, No.4, July, pp.1155-1162

Combacau M., Courvoiser M. (1991) Process Failure Diagnosis in FMS Real Time Control: an Approach Combining Rule Systems and Petri Nets, The 2nd Annual Conference in AI, Simulation and Planning in High Autonomy Systems, 1-2 April, Cocoa Beach, USA, pp.174-180

Guida M., Rossett V., Rosazza M., Wohlgemuth W., Wheeler S. (1992) DIREK: a Multi-model Real Time Diagnostic System, 8th CIM-Europe Annual Conference, Birmingham, U.K., 27-29 May, pp.420-430

McFarlane D., Marett B., Elsley G., Jarvis D. (1995) Application of Holonic Methodologies to Problem Diagnosis in a Steel Rod Mill, International Conference on Systems, Man and Cybernetics, Intelligent Systems for 21st Century, 22-25 October, Vancouver, Canada, pp.940-945

Batty D., Kamel M.S. (1995), Automating Knowledge Acquisition: A Propositional Approach to Representing Expertise as an Alternative to Repertory Grid Technique, IEEE Transations on Knowledge and Data Engineering, Vol.7, No.1, February, pp.53-67

Shankararaman V., Lee B.S. (1994) Knowledge-Based Safety Training System (KBSTS) - A Prototype Implementation, Computers in Industry, Vol.25, pp.145-157

Boy G. A. (1987) Operator Assistant Systems, Int. J. Man-Machine Studies, Vol.27, Nos.5&6, pp.541-554

Mayer, R., Menzel, C. Benjamin, P., Painter M. (1992) "IDEF3 Process Description Capture Method Report," Technical Report AL-TR-1992-0057, Wright Patterson Air Force Base, Ohio, U.S.A.

Menzel, C, Mayer, R.J. (1998) The IDEF Family of Languages. in Handbook on Architectures of Information Systems. Bernus,P., Mertins, K. and Schmidt, G. (Eds). Berlin : Springer, pp 209-242

Menzel, C., and Mayer, R. (1996) Situations and Processes. Concurrent Engineering: Research and Applications, Vol. 4, No. 3, pp.229-246

Mo J.P.T., Nemes L. (1998b). Framework for automatic creation of remote operations support systems, IFIP TC5 WG5.3/5.7 Third International Conference on the Design of Information Infrastructure Systems for Manufacturing (DIISM '98), May 18-20, Fort Worth, Texas, U.S.A., pp.93-104

Mo J.P.T., Law R.T.C. (2000). Diagnostics Modelling and Web-based Delivery, Proceedings of 8^{th} International Conference on Manufacturing Engineering, 27-30 August, Sydney, Australia, Vol.1, pp.523-527

KBSI (1995). ProSim^{TM} (1995). Automatic Process Modelling for Windows, User's Manual and Reference Guide, Ver.2.1, Knowledge Based Systems, Inc., College Station, Texas

Giarratano J.C. (1993). CLIPS User's Guide, Ver.6.0, NASA Lyndon B. Johnson Space Center, Information Systems Directorate Software Technology Branch, 1993

Mo J.P.T. (2000). Automated Management of Quality Control System for Network Enterprises, Global Engineering, Manufacturing and Enterprise Networks (Proceedings of 4th International Conference on Design of Information Infrastructure Systems for Manufacturing), 15-17 November, 2000, Melbourne, pp.228-235

Kamio Y., Kasai F., Kimura T., Fukuda Y., Hartel I., Zhou M. (2002). Remote Maintenance Support in Virtual Service Enterprises, Proceedings of 5th International Conference on Design of Information Infrastructure Systems for Manufacturing, 18-20 November, 2002, Osaka, pp.233-240

22

THE USE OF GERAM TO SUPPORT SMES DEVELOPMENT IN MEXICO

A. Molina and R. Carrasco

CSIM-ITESM**

22.1 Introduction

This case study describes how the GERAM[3] has been used to support the design, development and operations of SMEs[4] in Mexico. The use of GERAM is described using the particular case study of a plastic products company.
SMEs represent 99% of enterprises in developed countries and from 95% to 99% of enterprises in Latin American countries. Their contribution to the Gross Internal Product is an average of 51% in developed countries and an average of 43 % in Latin America (Emilio and Zevallos, 1999). Therefore, the development of SMEs is critical for the economic development of the countries, and this is particularly true for Latin America.
A program for the development of SMEs has been created by CSIM-ITESM in order to support SME by improving their competitive position. The objectives of this program, named IMMPAC[5], are (Molina and González, 1997):

- introduce and transfer Enterprise Integration concepts in Mexican Micro and SMEs;
- develop a methodology in conformance with GERAM, in order to engineer Mexican enterprises;
- create SME enterprise models for different types of manufacturing and services companies, and

** Centro de Sistemas Integrado de Manufactura - Instituto Tecnologico y de Estudios Superiores de Monterrey (Integrated Manufacturing Systems Centre - Technological and Superior Studies Institute of Monterrey), http://csim.mty.itesm.mx/

[3] Generalised Enterprise Reference Architecture and Methodology, as described in Bernus and Nemes (1996); ISO/TC184 (1999)

[4] Small and Medium Enterprises

[5] Acronym in Spanish for Integration and Modernization of Small and Medium Enterprises to Achieve Competitiveness

P. Bernus et al. (eds.), *Handbook on Enterprise Architecture*

- contribute to the Enterprise Integration research community with new developments and results.

The use of GERAM concepts in this program has been key for the success of the implementation of core concepts in SMEs. GERAM was selected to select, customise and formalise the modelling framework based on the following reasons:

- GERAM does not impose a set of tools and methods to the other reference models;
- As a generalised reference architecture, GERAM possesses the expresiveness necessary to contain other (more specialised) reference architectures.
- GERAM is part of ISO 15704 (Annex A) which is an important standard in the Enterprise Integration field.
- GERAM covers the full life cycle of enterprise entities, in this case SMEs.

GERAM uses three main concepts to guide the integration of the enterprise elements ((Bernus and Nemes , 1996; ISO/TC184 , 1999)):

- Entity Life Cycle;
- Modelling Views;
- Instantiation.

This Chapter contains a description and discussion of how to use these three major concepts in the development of SMEs.

22.2 The Use of the Concept of Life Cycle

22.2.1 Background

The enterprise *life cycle* describes the phases (or stages[6]) that an entity may go through its life cycle. These phases are the same for every entity so they can be captured in a graphical manner and serve as reference for the creation of a future entity (Williams , 1994). The entity can be an enterprise, a project, a product, etc. GERAM identifies eight phases to cover the complete life cycle of an entity. These phases are (refer Chapter 2 and (ISO/TC184 , 1999)):

- Identification: The set of activities that identify the contents of the particular entity under consideration in terms of its boundaries and its relation to its internal and external environments. These activities include the identification of the existence and nature of a need for a particular entity.

[6] while stages and phases are used interchangeably here, traditionally 'stage' implies a temporal dimension while phase does not. Hence, 'phase' would be more suitable to describe entity life cycle, while stage would be best used to describe its life *history*(the succession of life cycle phases through which an entity has gone-, or is reasonably expected to go through, during its existence).

- Concept: Activities needed to develop the concepts of the underlying entity. Includes mission, vision, values and strategies, business plans, policies and so forth.
- Requirements: Activities needed to develop descriptions of operational requirements of the enterprise entity, it's relevant processes and the collection of all it's functional, behavioral, informational, and capability needs.
- Preliminary Design: Overall enterprise specifications (sufficient to obtain approximate costs and management approval of the ongoing change project).
- Detailed Design: Major design work necessary for the complete system design suitable for building of the final physical system.
- Implementation: Activities that define all those tasks that must be carried out to build or rebuild the entity.
- Operation. The activities of the entity that are needed during its operation for producing the customer product or service which is its special mission along with all tasks needed for monitoring, controlling, and evaluating the operation.
- Decommissioning. These activities are needed for re-missioning, retraining, redesign, recycling, preservation, transfer, disbanding, disassembly, or disposal of all or part of the entity at the end of its useful life in operation.

The Life Cycle Diagram in GERA[7] describes all the pertinent activities needed to engineer an enterprise from its concept to its decommission. A detailed description of how all the concepts of GERA are used to support the design, development and operation of a SME is given in the following sections.

22.2.2 The Design and Creation of a SME

Plastipart is a medium size enterprise[8]. The company was created in 1983 by two brothers, Juan and Francisco Garza. This company employs 180 people. The enterprise is divided into five departments: production, engineering, quality, marketing, and administrative department. Plastipart produces plastic components mainly for the automotive industry. One of the competitive advantages of Plastipart is that it has the capability of producing its own injection molds. Ford is the main customer of this enterprise and purchases from Plastipart small components, such as knobs and buttons for the interior of its vehicles. Customers prefer Plastipart because this company supplies components of excellent quality, in a short time, and at a competitive price.
The agility of the company has also been demonstrated by its capability to develop new products for its customers in short periods of time. One of the factors that contribute to the quick development of new products is the fact

[7] Generalised Enterprise Reference Architecture, a component of the GERAM (refer Chapter 2 for details)

[8] The name of the company and some characteristics of the processes and parts described have been changed due to reasons of confidentiality.

that Plastipart mastered the application of the Life Cycle Concept to new products. This technique has permitted all the departments to work in an integrated and efficient way to develop a product that satisfies all customer requirements and obtain the desired profit. As a matter of fact, the Life Cycle- and many other GERAM concepts have been present in Plastipart from the beginning of this business. The Garza brothers employed the GERAM concepts to engineer, implement, and operate Plastipart. Figure 22.1 shows the life cycle phases for Plastipart with a description of the kind of activities executed during the life history of Plastipart.

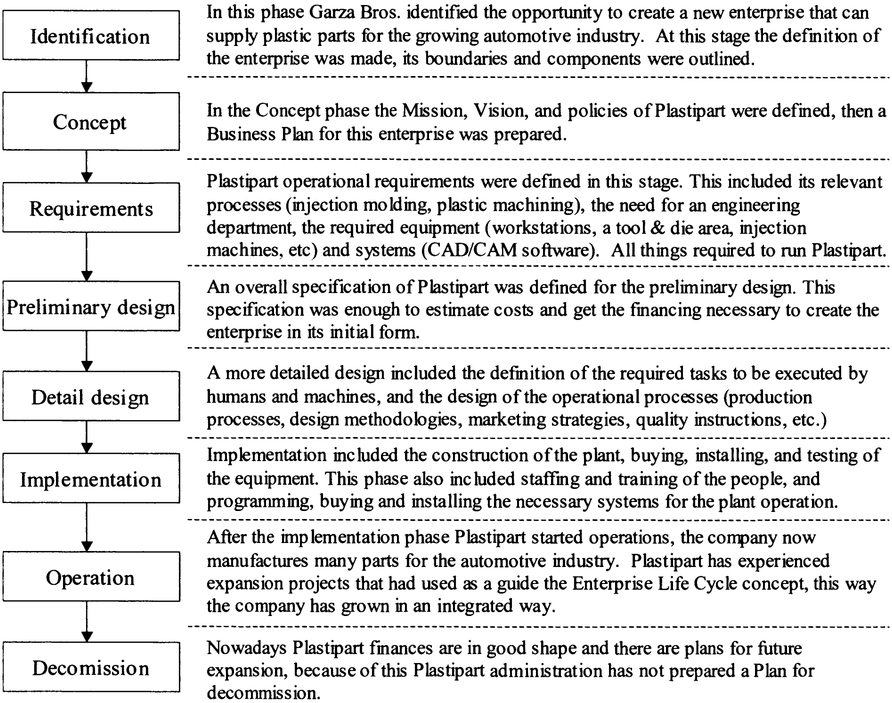


Fig. 22.1. Plastipart Life Cycle

22.2.3 Development of a New Business Opportunity Using the Life Cycle Approach

Ford Motor Company was planning to change the design of the shift knob for some of its models, so it requested Plastipart to develop the process for manufacturing this part. To guide the development of this new product, Plastipart

uses the same life cycle concept employed in its own development, however applied to the envisaged product. In the operation phase, the diagram represents all the activities that the enterprise executes so as to attain its objectives. One of these activities is the development of new products. Ford's requirement for a new shift knob has triggered the execution of this activity.
Therefore, Plastipart identified the need to design and manufacture a new product, and a new life cycle diagram was launched (refer Fig. 22.2). As previously mentioned, a life cycle diagram represents the phases through the life of the new product, in this case the shift knob. As a new product, the shift knob will have a life cycle related to the life cycle of Plastipart, as shown in Table 22.1.

Table 22.1. Life Cycle Phases for Ford's new product (shift knob)

Phases	Description
Identification	Ford identifies the need for a new shift knob for its vehicles. Plastipart, as key supplier for Ford, will participate in this new product development.
Concept	Outline objective of the new product (aesthetic, functional, etc).
Requirements	The specific functions, and standards that the new product has to comply with are stated and translated into product specifications.
Preliminary Design	Plastipart analyses product specification, estimates costs, and delivers a quotation for the new part.
Detailed Design	Once the order for the new product is received, Plastipart elaborates a detailed design that includes the design of the process (temperature, pressure,etc), the design of the mold, and the design of the piece. The deliverables of this activity are drawings and work instructions to manufacture the new part.
Implementation	This phase includes activities like the construction of the injection mold, run tests with the new mold, and prototypes of the new part. Then the volume of units required by Ford is manufactured and delivered.
Operation	In the operation phase the new parts are assembled in Ford's vehicles, and put in service.
Decommission	After the part is no longer in service, the plastic material can be recycled.

In order to better carry out the activities involved in the detailed design of the product, several viewpoints may be used to described how a specific

activity must be undertaken (such as e.g. the design of the injection mold). The various points of view are employed to analyse the design of the mold in order to establish what are the inputs/outputs of the activities involved, which components of the enterprise execute these activities and which tools are needed to support them.

The Function view represents the main activity, i.e. the design of the injection mold. The Organisation view shows that the responsible entity for the execution of this function is the Engineering Department. The Resource view shows the necessary resources for the design of the mold. These resources may be human and/or technological. In this case, the resources comprise Juan Ramos (Design Engineer, human), a Computer Aided Design System (CAD - non-human, software), and the workstation used to run this application (non-human, hardware). Finally, the Information, or Data view represents the informational inputs and outputs of the design of the mold.

The inputs are the design criteria in the form of documented rules and procedures. The outputs are drawings and work instructions to construct the injection mold. All the other activities necessary to manufacture the new product can be mapped within the four views in the same way. As previously mentioned, the life cycle diagram is a useful tool for analysis and planning of an entity (be it a product, project, enterprise, etc.). In the case of an enterprise, the life cycle diagram can provide information about the activities executed in each phase, and with the aid of the modelling views the flow of materials, information, and control between activities can be represented.

22.3 Organizing SMEs Using GERA (Generic Enterprise Reference Architecture)

The main component of GERAM is GERA, which is a generalised[9] life cycle architecture[10]. GERA is composed of three kinds of concepts (refer (ISO/TC184/SC5/WG1, 1999) and Chapter 2), which are as follows:

- Human oriented concepts, covering human aspects like human roles and the way human interact with other humans and technology.
- Process oriented concepts, dealing with enterprise operations and include entity types, entity life cycle, and modelling views.

[9] generalised in this context means that GERA is describes *types of* artefacts rather than the artefacts themselves (in an Object-Oriented analogy, the same way a class diagram contains *classes*, or *types of* objects rather than the objects themselves). E.g. the Product entity describes a type of product, of which many instances are produced. However, some entities have only one instance, for example the company Plastipart is such an entity.

[10] a life cycle (or 'type 2') architecture describes the possible phases and artefacts involved during the existence of a system, rather than being just a snapshot of the structure of an artefact at a given point in time. For further details on type 1 and 2 architectures please refer to Chapter 2.

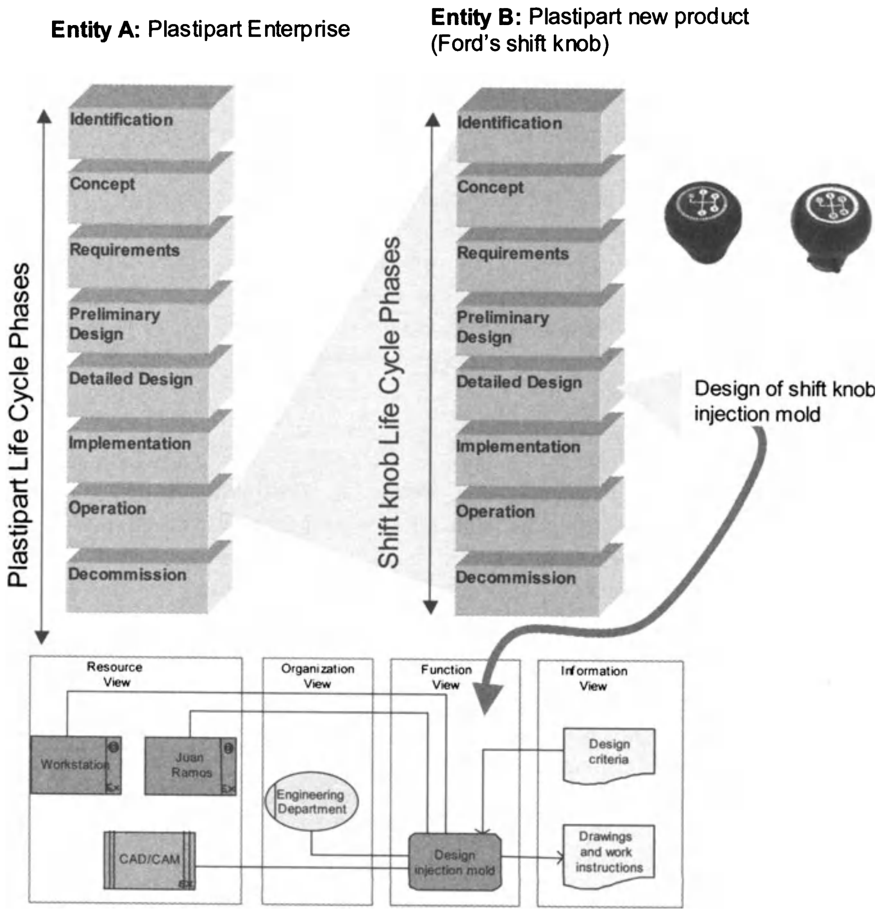


Fig. 22.2. Plastipart Life Cycle

- Technology oriented concepts, dealing with infrastructures necessary to support processes.

22.3.1 The Use of Human Oriented Concepts

GERAM takes into account the human part of the enterprise. It is now widely accepted that the most important part of the organisations is its human resources. GERA employs a dedicated concept to define the characteristics of the individuals that work in the enterprise and the kind of work that they do. This concept is the 'human role'. Human roles define the activities of people within the organisation, and in addition they also describe the profile (skills and characteristics) of the individual. GERA is also able to describe the way humans interact with other humans, or with the technology (equipment or

systems). This can be accomplished through modelling languages, in which the activities of humans and the technological tools used to perform activities are represented using diagrams. The diagrams are graphical descriptions of the transformation of information and/or materials, employing human and technological[11] resources.
Garza Bros. employed the GERAM concepts of human role(s) to engineer Plastipart. During the execution of the preliminary design and detailed design phases, Plastipart's processes and organisation were defined. The organisation's definition required the creation of human roles that describe the activities and profiles of humans in Plastipart. The human roles for the Managers of Plastipart departments are presented below:

Engineering Manager

Skills: This person has to be a leader, innovator, and open minded;
Knowledge: Manufacturing and maintenance background (plastic molding), CAD/CAM proven experience, and Six-Sigma (Methodology to reduce the number of defects per million in a product or service) training and application knowledge;
Activities:

- administrate Engineering Department,
- provide basic training to maintenance technicians and designers;
- plan and lead the implementation of technological tools (CAD/CAM);
- explore new, more efficient processes to manufacture Plastipart components and
- support the other departments with any quality or production capacity problem.

Quality Manager

Skills: This person has to be customer oriented and have excellent communication skills;
Knowledge: Molding process background, domain quality basic tools, Six Sigma and ISO9000 (an international standard for quality systems) literacy;
Activities:

- periodically audit office and operative processes and products;
- elaborate diagnostics and corrective actions for quality issues;
- be responsible for ISO9000 procedures documentation;
- lead Six-Sigma improvement projects and
- serve as Plastipart interface with the customer in any quality issue.

[11] the term 'technology' is used in this Chapter with the meaning of non-human, software and hardware resources.

Production Manager

Skills: Strong leadership, open minded, experience managing operative personnel, proactive, and team-work oriented, oriented to productivity and safety.
Knowledge: Molding process background, tool & die fabrication experience, statistical tools literate, domain safety procedures.
Activities:

- manage production activities;
- monitor production performance and requirements;
- report productive indicators and lead the implementation of projects (quality circles) to improve these indicators;
- lead safety training.

Marketing Manager

Skills: Excellent communication skills, customer oriented, proactive, and innovative person.
Knowledge: National and International automotive market knowledge, domain of legal terms and conditions of contracts in this kind of industry, sales estimation, competitive intelligence.
Activities:

- responsible for order fulfilment;
- search and consolidate business opportunities;
- responsible of customer service and
- planning of new product development.

As shown above, the human roles for Plastipart managers describe the kind of activities to be done by these individuals and the skills and knowledge necessary to play each role.
The modelling of the interaction of humans with other humans and with technology is presented later on using modelling languages.

22.3.2 The Use of Process Oriented Concepts

The objective of these concepts is to describe the business processes in the enterprise, capturing both their functionality (i.e., *what* has to be done) and their behavior (that is, *when* things are done and *in which sequence*). Three process oriented concepts are used:

- Entity types
- Entity Life Cycle
- Modelling Views

Entity types: This concept refers to the fact that GERA can be used to analyse many *kinds of* entities, for example an enterprise, a project, a product, etc .
Entity Life Cycle: Any entity has a beginning and an end to its useful life; the types of activities performed in these phases are described in this concept. The principal contribution of the life cycle is to provide a guide or methodology to create an entity in the future.
Modelling Views: To engineer or re-engineer the processes of an enterprise it is necessary to represent these processes, together with their inputs and outputs, and the resources used to accomplish them. As previously shown, this representation employs four modelling views: function, organisation, information, and resources.

22.3.3 The Use of Technology Oriented Concepts

"Technology oriented concepts have to provide descriptions of the technology involved in both the enterprise operation and the enterprise engineering efforts" (IFIP-IFAC, 1999). The descriptions are basically resources models and descriptions of the enterprise engineering tools used to develop and update these models.
Garza Bros. used technology oriented concepts to design the operations of Plastipart. Models were constructed for the different activities necessary to run the enterprise. For example, an overall description of the molding process of a plastic part was modelled. To accomplish this, Garza Bros. first made a list of the steps required to inject a plastic part. After the list was completed, they defined the people involved in the execution of each step. Next, they listed the resources required to inject the part and the information necessary to complete these task. The following step was to create graphical models of the steps, players (people), resources, and information.
Garza Bros. had alternatives to create the models: they could have drawn them manually and then document them, or they could have used specialised software for engineering business processes. They have selected the second option, because in that way the models could be stored in a database and be reused in the future. In addition model maintenance is easier using business process modelling tools than for models created manually. The process to create these models is shown in Fig. 22.3. This figure shows the list elaborated by Garza Bros., which contains the pertinent steps to perform the injection molding of a plastic part. It also shows the people that take part in this activity, the resources used, and the necessary information. All these elements (people, functions, information and resources) form the molding process. Specialised modelling software was used to represent this process. The output of this modelling process is the diagram shown at the bottom of Fig. 22.3. The diagram represents the four mentioned elements of the molding process. Functions are represented by a square with a number, where the number corresponds to the sequence of steps. Names of people (human roles) have been abbreviated in the diagram (e.g. SUP stands for Supervisor, OP 1 stands for

Operator 1, OP 2 for Operator 2, etc). Letters are used to represent the information required (for example letter "A" means "Material data"). Finally the resources are also represented in the figure using abbreviations (e.g.CAR-Material car, HUM-Humidity Detector, INJ-Injection Machine, DRY- Plastic Dryer).

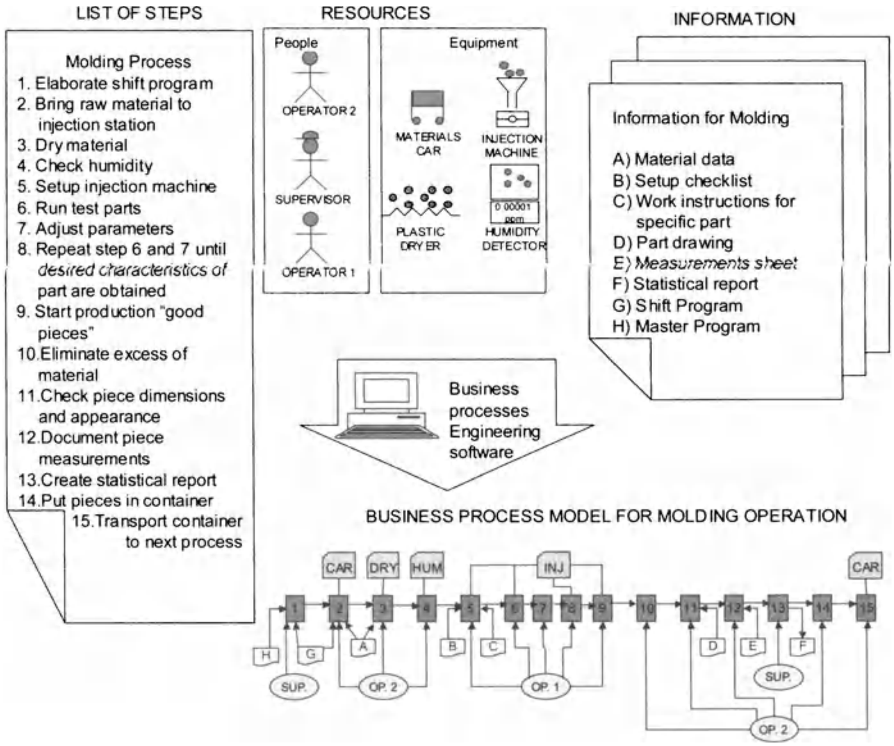


Fig. 22.3. Modelling of the Molding process

The diagram represents all the operations, who performs them and what is used to perform each operation (information or resources). For example, at the top-left corner of the diagram it can be seen that the first function is to "elaborate the shift program" [1] (the production program for that shift) and that the responsible performer of this action is the supervisor [SUP]. The supervisor needs the master program [H] to elaborate the shift program [G]. The next step is to "bring material to injection station " [2], which is performed by OPERATOR 2 [OP. 2]. The resource required for this function is the material car [CAR], and the information needed is the material data [A]. Each of the steps is described in a similar fashion. Modelling provides a way to represent complex functions and relations in a simple manner, and it

has been found that if the modelling language(s) used are mastered by the people within the enterprise, the communication of relevant information will be easy, precise, and quick.

22.4 Documentation of Processes and Methods in SMEs Using EEMs (Enterprise Engineering Methodologies)

"Enterprise engineering methodologies describe the processes of enterprise integration. Their scope is defined in the GERA life-cycle concept. It provides methods of progression for every type of life-cycle activity. The methodology may be specifically oriented to the type of enterprise or entity under consideration" (IFIP-IFAC, 1999). For example, in the case of SMEs, methodologies that put more emphasis on the requirements of this type of enterprise can be created.

Plastipart is an SME that supplies products to a very demanding customer – a car manufacturer with very strict policies about quality, cost and lead times. For this reason, to be competitive, Plastipart has developed specific methodologies to comply with the requirements of its customers. One of these methodologies is its product design methodology. Plastipart has to develop new products in short cycle times, which designs have to comply with all Mexican and US safety and environmental regulations. Materials and process cost are critical for these kinds of products. All these factors have contributed to the creation of a design methodology that can be used in general by all SME suppliers of automotive components.

Figure 22.4 shows the design methodology used at Plastipart. It is an Enterprise Engineering Methodology in the sense that it establishes a set of steps to perform a design process. The general case can be used to engineer the design process of SME suppliers of automotive components.

Referring to Fig. 22.4 :

- EPA and SEMARNAP are environmental regulation agencies;
- Finite Element Analysis is a technique that uses computational algorithms to simulate and calculate physical phenomena.
- Fatigue is the damage a material component suffers after repeated cycles of operation.

22.4.1 Using EMLs (Enterprise Modelling Languages) to Formalise Process Documentation

EMLs define the generic modelling constructs to "describe and model human roles, operational processes and their functional contents as well as the supporting information, office and production technologies" (IFIP-IFAC, 1999).

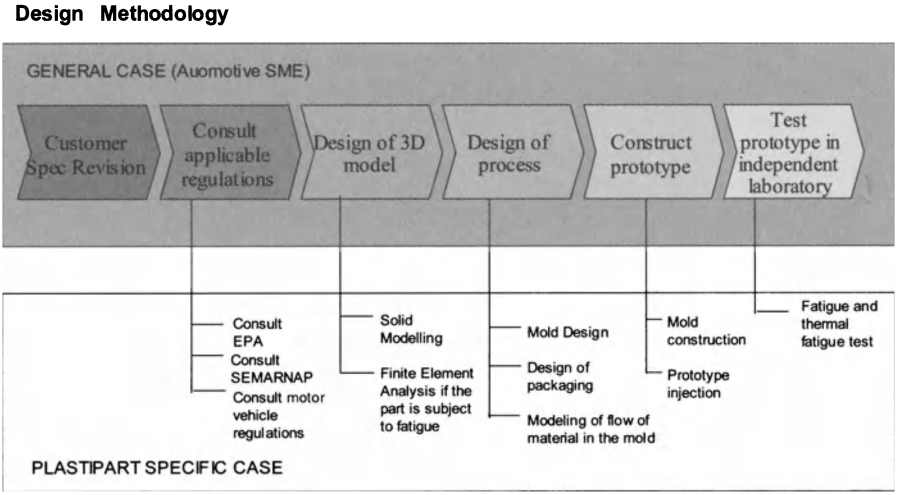


Fig. 22.4. Design Methodology for automotive component

Modelling languages are used to engineer the business process of the SMEs. This is done representing the business processes, their inputs and outputs and the agents (human/technological) that execute the processes using up to four modelling views. These views are: function, information, organisation, and resources.
In this case the **Functional View** was constructed using the e-EPC (Event driven Process Chains) modelling language implemented in the ARIS toolset (IDS Scheer, 2001). In this language, a process is represented in a diagram as a sequence of activities (functions), with each activity being triggered by an event and also producing an event in turn. This kind of diagram links views called function, data, and organisation view in ARIS. The elements of an e-EPC diagram are:

- functions (activities that are executed to reach business goals),
- events (something that happens, triggering functions and being produced by functions),
- organisational units (entities that are part of the organisation),
- data objects (documents, drawings, electronic data, etc),
- Links (AND, OR connectors to link events or functions),
- Process Interface (links between processes).

The engineering manager of Plastipart wanted to construct a model of the Design process of plastic parts. This was done with the objective of explaining the design process to two new designers. The model is only a high level representation – more detailed models can be created using this representation. The resulting model is shown in Fig. 22.5.

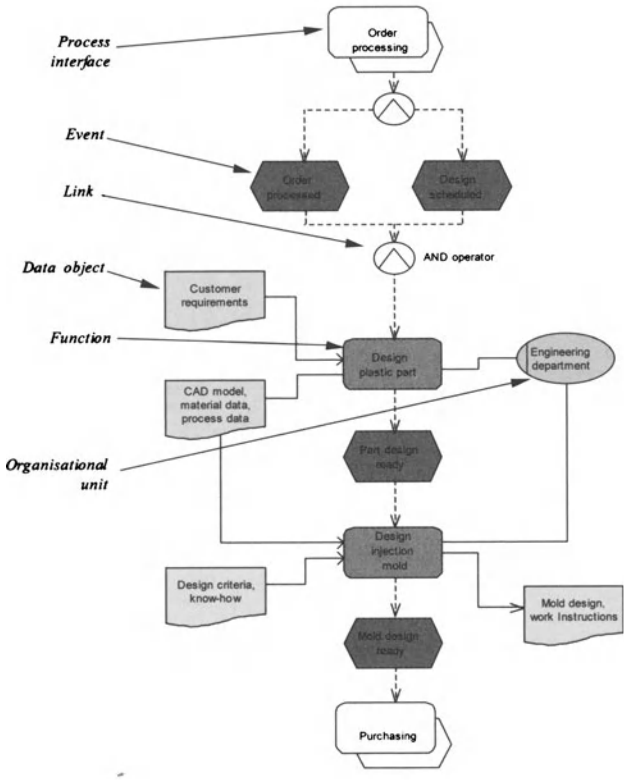


Fig. 22.5. Plastic part and injection mold design e-EPC

In the model a previous process is represented by the process interface "Order processing". Order Processing involves all the paper work done by the marketing department to process an order. The execution of this process produces two events: one signifying that the order has been processed, and the other that the design for that order has been scheduled. The presence of both events together triggers the execution of the function "Design plastic part". The Inputs to the "Design plastic part" activity are 'customer requirements' (which is a data object in the form of a specification document). The outputs of this function are the CAD model, material data (in this case, type of plastic), and process data (in this case injection time, pressure, temperature, and material quantity). This design task is executed by the Engineering Department (represented in the diagram by the ellipse), and it produces the event "Part design ready". This event triggers the execution of the "Design injection mold" function. The inputs of the function are the CAD model, material data, and process data. The outputs are the mold design and work instructions (a process plan describing the steps to fabricate the part). The "Design injection mold" function is also executed by the Engineering Department, and it

produces the event "Mold design ready". Finally, the Design tasks are linked to the next process (Purchasing) by a process interface.
In the **Organisation View** simple Organisational Charts were used as a Modelling language. Organisational Charts are diagrams in which the various components of an organisation are represented. These components are modelled in the form of *organisational units.* An organisational unit can be a group of enterprises, a single enterprise, a division of the enterprise, a department or group within the enterprise, or an employee of the enterprise. Therefore, organisational units can be decomposed down to the level of the individual to represent the human roles within the organisation. The organisational chart produced is shown in Fig. 22.6. The Engineering Manager wanted to show to the new designers the organisational structure of Plastipart and especially the structure of the Engineering Department. For this purpose, the Engineering Manager constructed a model of the organisation with a focus on the Engineering Department (Fig. 22.6).

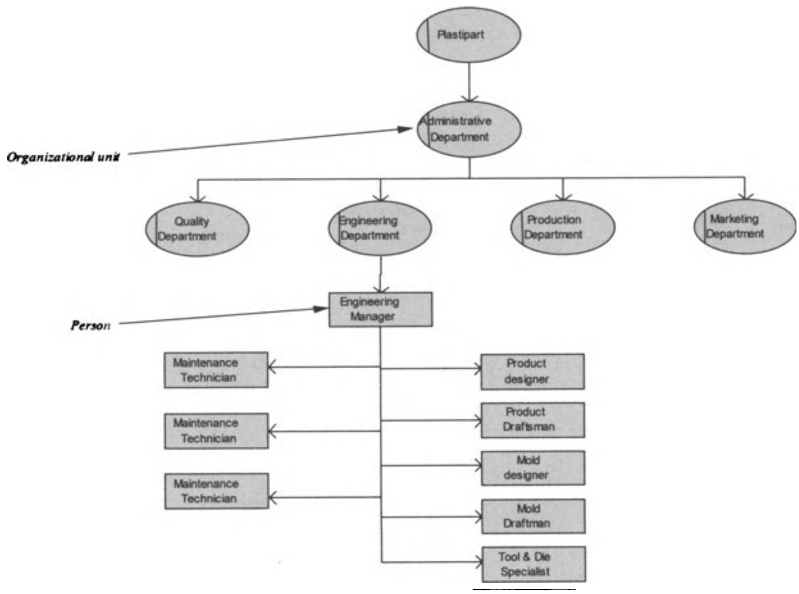


Fig. 22.6. Plastipart organisational chart, focus on Engineering department[13]

This model shows that Plastipart is composed of five departments, represented as organisational units. The model also shows that the Administrative

13 in the subsequent Figures within this Chapter, text in italics and the associated arrows are just explanation aids and not part of the model

Department is hierarchically superior to the other four departments[14]. The head of the Engineering Department is the Engineering Manager, represented by a person (rectangle) in the diagram. This manager is in charge of a product designer and draftsman, the mold designer and draftsman, and three maintenance technicians.

In this case the **Information View** models were created using Function Allocation Diagrams (FADs) as a modelling language. This model represents the information exchange among processes. FADs also represent the information technology resources that support the exchange of data, and detail the kind of information exchanged. Modelling objects in FADs are Functions, Data objects (documents, and electronic data), Organisational units, and Applications (technological tools like software).

A manufacturing engineer (part of the Production team) concluded that the way the Engineering Department delivers fabrication information to Production is not efficient. This engineer believed (and had data to support that belief) that the fabrication information is not always updated, and sometimes is not complete. Hence, he wanted to propose the implementation of direct interfaces between the CAD design software and the machining center and injection machines for mold and plastic parts fabrication.

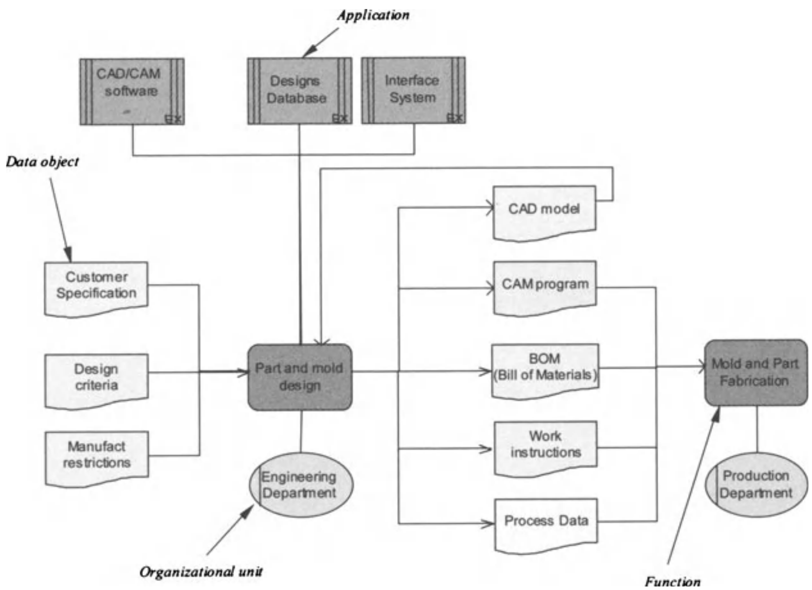


Fig. 22.7. FAD with proposed modification to information delivery in Plastipart

[14] Editor's note: the exact nature of hierarchical relations between management roles could have been represented using Grai Grids (refer to Chapter 12).

For this purpose, Plastipart subcontracted an Information Technology specialist to analyse this situation. This person, assisted by the Engineering department constructed a FAD of the proposed Interfaces for the design process. Figure 22.7 describes two processes. One is "part and mold design" and the other is "mold and part fabrication". The inputs to the first process are the data objects: customer specifications, design criteria, and manufacturing restrictions. The outputs of the design are: CAD (Computer Aided Design) model, CAM (Computer Aided Manufacturing) program, BOM (Bill of Materials), work instructions, and process data. Notice that CAD model is an output of design but also and input to this process, because it is used to generate the CAM program (instructions for the fabrication of the mold in the machining center). The design process is supported by three applications, the first application is the CAD/CAM software required for part design and translates the part information into CNC (Computer Numeric Control) code. The designs database stores all the design information for the plastic part and injection mold. The Interface system provides connectivity from the CAM software to the machining center, and injection machines. Finally we can observe that the design function is executed by the Engineering Department, and the fabrication function is executed by the Production Department.

The **Resource View** used Technical Resource Diagrams as a modelling language. These models were used to represent the human and technological resources of the organisation. The objects in these diagrams are Technical Resources, and Locations. A Technical Resource can represent a human resource (a person playing a human role, requiring specific skills and responsibilities), ICTs (Information and Communication Technologies), and equipment (drilling machines, a fax, a computer, a forklift, etc). Locations are the places where the resources are available.

Plastipart supposed that the Production Manager will want to construct a model that represents Production Resources in order to be able to clearly see the possibilities and effects of resource re-allocation. E.g., in the recent months the demand of mirror frames for General Motors (GM) vehicles have decreased. At the same time the demand of Shift knobs and glass holders for Ford vehicles have increased. The Production Manager would want to reallocate resources from the mirror frame production line to the shift knob and glass holder line. To decide which is the best relocation strategy the Production Resources model can be used. Figure 22.8 shows a preliminary model focused on GM line resources. In this diagram all the Production Resources are located at the Plastipart facility (location).

The preliminary model is focused on the GM line. The technical Resources of the GM line can be divided into three categories:

- Human (Supervisors and Operators),
- Applications (Computer-aided Process Planning – CAPP software),
- Equipment (Injection machines and materials cars).

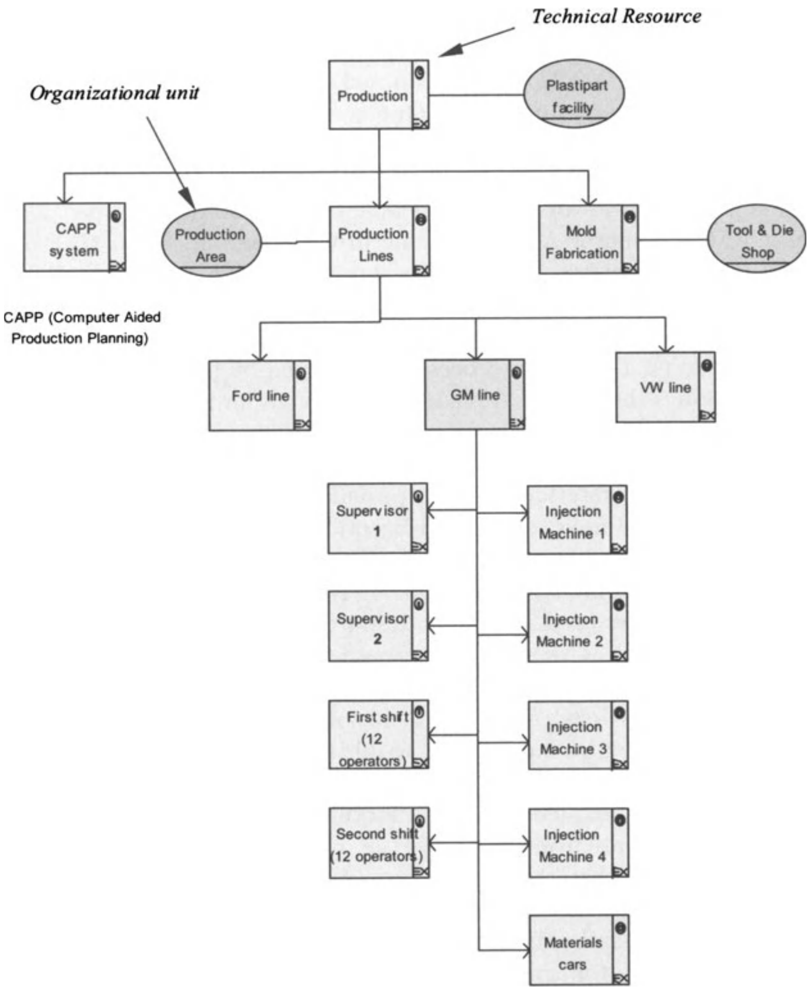


Fig. 22.8. Plastipart Production Resources, focus on GM line.

The next step for the Production Manager is to model the Ford line, and then check which resources can be transferred form GM to Ford line according to the demand.

22.4.2 EETs (Enterprise Engineering Tools) for Electronic Documentation in SMEs

Enterprise Engineering Tools (EETs) are applications that assist the enterprise designer in planning, analysing and modelling the behavior of processes within the enterprise. If the enterprise is a new entity (i.e. still to be created),

then EETs help in designing its components. If the enterprise already exists, then EETs can facilitate enterprise re-engineering or process improvement. For example, EETs can help management take decisions by modelling and simulating the enterprise's processes. In the present case study the EET used is a static model, because its objective is to serve as a guide, for replication in CSIM's Virtual Industry Cluster (VIC).
The EET used in this particular case study was ARIS (Architecture of Information Systems) (Scheer, 1998, 1999). ARIS is based on a modelling framework for the development of business process models. Hence, it contains all the basic features to describe the business processes of an enterprise. The model of an enterprise is very complex; therefore, the ARIS architecture divides it into individual views to reduce its apparent complexity [15]. Due to this division, the models of the individual views can be analysed by methods suitable for the particular view without paying attention to the relationships of the model to other views (Scheer, 1999). The ARIS modelling framework is supported by the EET called the ARIS Toolset.
"The ARIS Toolset and its add-on components enable the enterprise-wide and global definition and design of business processes as well as their analysis and optimisation" (IDS Scheer, 2001). This software uses a graphical modelling system, a library of modelling languages, and a database to store the various developed models[16]. The modelling work presented in this case uses four different modelling Languages:

- e-EPCs (Event Driven Process Chains) for the Function View;
- FADs (Function Allocation Diagrams) for the Information View;
- Organisational Charts for the Organisation View;
- Technical Resource Diagrams for the Resource View.

Garza Bros. decided to use the four modelling views (function, information, organisation, and resource) to design the processes of Plastipart. To do this, they adopted the ARIS Toolset as their Enterprise Engineering Tool. Figure 22.9 shows an ARIS Toolset window representing an EPC (Event driven processes chain) that models a Plastipart design process for a particular order (taking place within Plastipart's Operation life cycle phase).

22.4.3 Use of GEMCs (Generic Enterprise Modelling Concepts) to Document Key Terminology in SMEs

GEMCs (Generic Enterprise Modelling Concepts) are descriptions of modelling concepts employed in modelling languages. In the case study, GEMCs will have the form of Glossaries, in which the terms used in GERAM and

[15] for the mapping of the ARIS modelling framework onto the GERA modelling framework refer to Chapter 3

[16] for a brief presentation of the ARIS architecture framework and ARIS modelling tool refer to Chapter 3.

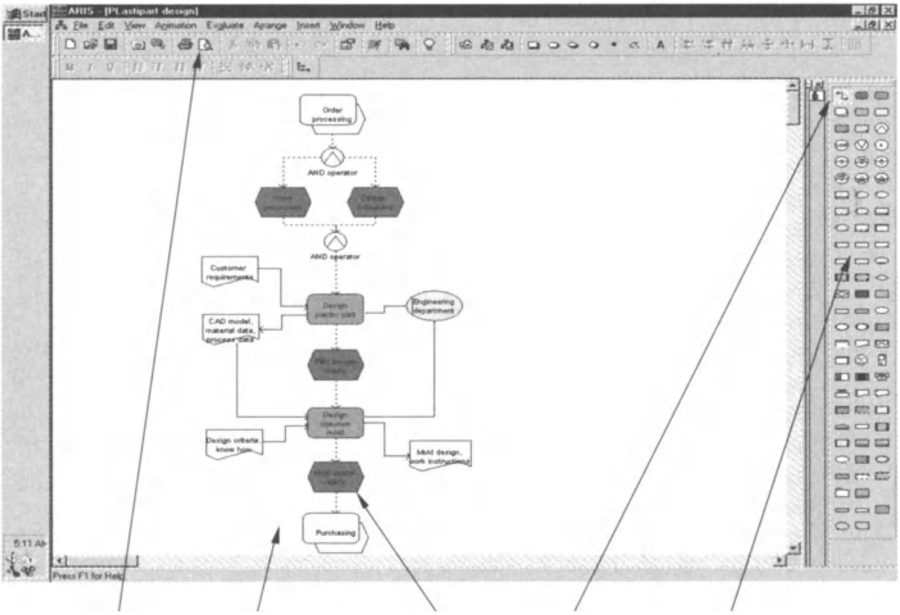


Fig. 22.9. Plastipart design process modelled with ARIS

in the modelling languages will be explained. Such terms, used within the Plastipart modelling processes, could be for example: the human roles of the company, specific processes, technical terms, etc.
In addition to these generic terms (terminology) the Glossary includes the definition of more concrete concepts as well. For example, the Plastipart Glossary contains definitions for the terms below:

- Engineering Manager (human role): Person responsible for the Administration of Engineering Department, basic training to maintenance technicians and designers, planning and leading the implementation of technological tools (CAD/CAM), exploring new and more efficient processes to manufacture Plastipart components and support the other departments with any quality or production capacity problem.
- Plastic Drying (Specific process): Operation made prior to Plastic injection. Plastic raw material has the form of small flakes. These small pieces contain a high percentage of humidity, depending on environment conditions. To perform the injection operation, this material has to be dried in an special dryer machine next to the injection machine. Before using this raw material, its humidity has to be checked so as to ensure that the drying process was sufficient.

- Thermal fatigue (Technical term): Plastics are affected in a considerable way by temperature. In the case of automotive components for the interior of the car, the components are exposed to extreme high and low temperatures. This changes in temperature are thermal cycles that affect physical properties of plastics. This factor has to be taken into account in selecting the kind of material for the design of an automotive component.

22.5 Conclusion

There are many opportunities to institute the use of Enterprise Integration concepts in Mexican Companies, especially in Micro, Small and Medium Enterprises. The emergence of global markets and Mexican economic growth are pushing these firms to accept new concepts in order to improve their competitiveness. A program named IMMPAC has been created to introduce the concept of Enterprise Modelling and Integration in Micro and SMEs in Mexico. This project integrates training and consultancy programs, in order to support enterprise engineering projects. This program has been using the GERAM concepts in order to formalise the design, development and operation of SMEs. IMMPAC has been successfully applied to nearly 40 SMEs in Mexico, and it is part of the methodology of the new research project called Virtual Industry Cluster (targeted to introduce the concept of Virtual Enterprise in Mexican Micro and SMEs, in order to become more competitive in the NAFTA market (Molina and Flores, 1999)).

References

Bernus, P., Nemes, L., and Williams, T. J. (1996a). Architectures for Enterprise Integration.London: Chapman & Hall.

Bernus,P. and Nemes,L. (1996) A Framework to Define a Generic Enterprise Reference Architecture and Methodology. *Computer Integrated Manufacturing Systems* 9,3, pp 179-191

IDS Scheer. (2001). ARIS 6 - Collaborative Suite, [White Paper]. IDS Prof. Scheer GmbH. Available: www.ids-scheer.de [2002, Aug 2002].

Scheer, A.-W. (1998). ARIS-Business Process Modeling (2^{nd} ed.). Berlin: Springer-Verlag.

Scheer, A.-W. (1999). ARIS-Business Process Frameworks (3^{rd} ed.). Berlin: Springer-Verlag.

IFIP-IFAC Task Force (1999) GERAM: Generalised Enterprise Reference Architecture and Methodology, Version 1.6.3 (March 1999), www.cit.gu.edu.au/~bernus/

ISO/TC184/SC5/WG1. (1999). Annex A: GERAM. In ISO-1999 (Ed.), ISO/IS 15704: Industrial automation systems - Requirements for enterprise-reference architectures and methodologies.

Molina A. and D. González (1997). IMMPAC: A Methodology for the Implementation of Enterprise Integration Programmes in Mexican SMEs. In Enterprise Engineering and Integration: Building International Consensus, K. Kosanke and James G. Nell (Editors), Springer, pp. 431-438.

Molina A., and Flores M.(1999) A Virtual Enterprise in Mexico: From Concepts to Practice, Journal of Intelligent and Robotics Systems, 26: 289-302, 1999.

Williams T. J. (1994). Architectures for integrating manufacturing activities and enterprises, Computers in Industry, Volume 24 , p.p. 111-139.

Emilio G. Zevallos V. (1999) PyME ó empresa media, repensando conceptos. published in página de FUNDES http://www.FUNDES.org (July 1999)

Index